

Eindverslag Stage 3devo

Verwarmde behuizing voor het 3D-printen van HDPE en PP
door
Luca van Straaten (18073611)

Dit document is opgesteld voor de stage bij 3devo. Luca van Straaten is een student Elektrotechniek aan de Haagse Hogeschool te delft.

Datum: 19 januari 2022
Versie: 2.0

WIJZIGINGEN

Herziening	Datum	Auteur(s)	Beschrijving
V0.1	08.09.2021	Luca	Document aangemaakt
V0.2	29.10.2021	Luca	Eerste opzet
V1	17.11.2021	Luca	Eerste oplagen

VOORWOORD

Dit ontwerpdocument is geschreven als documentatie van het eerste stagetraject van Luca van Straaten.

Dank gaat naar 3devo voor de mogelijkheid om bij hun mijn eerste stage te doen.

Dit verslag is bedoeld voor mijn stage beoordeler om te beoordelen of ik voldoende heb gepresteerd tijdens mijn stage, voor 3devo en de mensen die daar werken en verder moeten bouwen op mijn werk en voor Nederlandstalige geïnteresseerden in het project. Een Engels vertaalde versie van dit verslag is momenteel niet beschikbaar.

Utrecht, November 2021
Luca van Straaten

INHOUDSOPGAVE

VOORWOORD	iii
LIJST VAN FIGUREN	vi
LIJST VAN AFKORTINGEN	vii
LIJST VAN BEGRIPPEN	viii
INLEIDING	ix
1 Wat is 3D-printen	ix
1.1 FDM 3D-printen	ix
1.2 3D-printer software	ix
1.3 3D-printer firmware	x
2 Wat zijn HDPE en PP	x
2.1 Waarom HDPE en PP	x
3 Bestanden downloaden	xi
1 ANALYSE VAN HET PROBLEEM	1
1.1 Waarom is een speciale 3D-printer nodig	1
1.1.1 Printen op een conventionele 3D-printer	1
2 EISEN VAN HET PROJECT	3
2.0.1 Randvoorwaarden	3
2.0.2 Functionele wensen	3
2.0.3 Gebruikerswensen	3
2.0.4 Ontwerpbeperkingen	3
3 PROBLEMEN	4
3.1 Smeltend plastic in de extruder	4
3.1.1 Oorzaak	4
3.2 Dubbelvouwend plastic in de extruder	4
3.2.1 Oorzaak	4
3.3 Kamer temperatuur overshoot	5
3.3.1 Oorzaak	5
4 MOGELIJKE OPLOSSINGEN	6
4.1 Extruder problemen	6
4.1.1 Verschillende soorten extruders	6
4.1.2 Zou een Bowden extruder de problemen oplossen	6
4.1.3 Zou een water gekoelde extruder de problemen oplossen	7
4.2 Temperatuur overshoot	7

Inhoudsopgave

5 DE GEKOZEN OPLOSSING	8
5.1 Extruder	8
5.2 Temperatuur overshoot	8
6 ONTWERP VAN DE OPLOSSING EN DE BENODIGDE ONDERDELEN	9
6.1 Hardware	9
6.1.1 Ge-3D-printe oplossingen	9
6.2 Electronica	10
6.3 Software	11
6.4 Firmware	11
7 ASSEMBLAGE	12
7.1 Assemblage van de printer	12
7.1.1 Electronich	12
7.2 firmware	12
8 TOETSING EINDRESULTAAT AAN DE HAND VAN DE EISEN	13
8.1 Veiligheid	13
8.2 Eenvoudigheid	13
8.3 Overig	13
9 CONCLUSIE EN AANBEVELINGEN	14
BIBLIOGRAFIE	15
APPENDICES	
BIJLAGE A INTERVISIE	17
BIJLAGE B STUDENT OVER DOCENTBEZOEK	19
BIJLAGE C BEDRIJF OVER STUDENT	21
BIJLAGE D STUDENT OVER BEDRIJF	23
BIJLAGE E VERANTWOORDING VAN DE BEHAALDE COMPETENTIES	25
BIJLAGE F KORTE BESCHRIJVING VAN DE BEDRIJFSORGANISATIE	27
BIJLAGE G OVERDRACHT DOCUMENT	29
BIJLAGE H CONFIGURATION	33
BIJLAGE I CONFIGURATION ADVANCED	93

LIJST VAN FIGUREN

1.1	PP bakjes van een oogheelkunde kliniek	1
1.2	Test print met HDPE op een conventionele printer	2
1.3	Test print met PP op een conventionele printer	2
4.1	Diagram van de twee soorten extruders die veel woorden gebruikt [1].	6
6.1	Render van het 3D ontwerp van de voetjes van de printer	9
6.2	Render van het 3D ontwerp van de afstandhouder van de printer	10
6.3	Render van het 3D ontwerp van het haakje van de printer	10
6.4	Render van het 3D ontwerp van de DIN-rail PCB houder	10
6.5	Render van het 3D ontwerp van de schakelaar houder op de DIN-rail	11
6.6	Render van het 3D ontwerp van het haakje van de C-14 houder op de DIN-rail	11

LIJST VAN AFKORTINGEN

2D Twee dimensionaal

3D Drie dimensionaal

3mf 3D Manufacturing Format (3D-productieformaat)

AM Additive manufacturing (Additieve productie)

f360 Fusion 360

FDM Gefuseerde depositiemodellering (Fused deposition modeling)

HDPE High-density polyethylene (Hogedichtheidpolyethaan)

PE polyethylene (polyetheen)

PID proportional–integral–derivative (proportioneel–integraal–derivaat)

PP Polypropylene (Polypropeen)

stl Standard Triangle Language (Standaard driehoekstaal)

LIJST VAN BEGRIPPEN

C++ programmeertaal gebaseerd op C

extruder Het gedeelte van een FDM 3d printer waar het filament mee door het hotend en de nozel word geperst

hotend Het gedeelte van een FDM 3d printer waar het filament smelt

INLEIDING

Waar de reden tot creatie van dit verslag worden blootgelegd, de opdracht en het probleem worden beschreven en achtergrondinformatie wordt gegeven.

De opdracht is om aanstuur elektronica te maken voor een 3D-printer die HDPE en PP moet kunnen printen. Deze printer is ongebruikelijk omdat deze een verwarde kamer krijgt die tot 160 °C moet kunnen komen. De hypothese is dat een verwarmde werkruimte van een dusdanig hoge temperatuur, de problemen oplost die worden bevonden met het printen van HDPE en PP met een conventionele printer.

1 WAT IS 3D-PRINTEN

3D-printen is een additieve productie (AM) methode. Dat betekent dat een product wordt opgebouwd vanaf nul, in tegenstelling tot subtractieve productie is er weinig of geen verlies van materiaal. Een andere vorm van additieve productie is bijvoorbeeld sputtigieten, maar hierbij zijn kostbare gietmallen nodig. 3D-printen vereist geen grote investering voor elk nieuw ontwerp, en is daarom uitermate geschikt om snel prototypen te maken. [2]

1.1 FDM 3D-PRINTEN

Er zijn verschillende vormen van 3D-printen, maar verreweg de meest populaire bij hobbyisten en kleinschalige bedrijven is FDM 3D-printen. FDM staat voor Fused deposition modeling, hierbij wordt steeds een laag materiaal op de vorige laag neergelegd (depositie) en daarop vast gesmolten (gefuseerd). Door herhaaldelijk laagjes op elkaar neer te leggen kan een 3D object worden opgebouwd. Het plastic wordt gesmolten in het hotend en wordt door de extruder uit het hotend geperst.

1.2 3D-PRINTER SOFTWARE

Er is verschillende software en firmware nodig om te kunnen 3d printen. Wat voor soorten software dat staat hier onder beschreven.

3D BESTANDEN

Software is nodig om een 3d object te ontwerpen, tijdens dit project is daarvoor f360 gebruikt. Ook kunnen bepaalde 3D objecten worden gedownload van [Thingiverse](#). Dit zijn meestal stl en 3mf bestanden.

Inleiding

SLICER

Gedurende dit project is de slicer *Prusaslicer* gebruikt, alle settings die gebruikt zijn om te slicen zijn op GitHub bijgehouden. Een slicer is software die een 3D bestand als een stl of 3mf, omzet in een hoop 2D lagen. Elke laag is gebruikelijk rond de 0.2 mm hoog. De printer *tekent* deze lagen en stapelt ze op elkaar, zo word een 3D object opgebouwd. In de slicer kan je veel dingen instellen om print kwaliteit of print snelheid te verbeteren. Instellingen zoals snelheid, temperatuur, flowrate en een hoop meer kun je tweaken voor een beter resultaat, deze instellingen zijn meestal printer afhankelijk.

1.3 3D-PRINTER FIRMWARE

Firmware is een specifieke klasse van computersoftware die de controle op laag niveau biedt voor de specifieke hardware van een apparaat. Voor minder complexe apparaten (zoals 3D-printers) kan firmware fungeren als het complete besturingssysteem van het apparaat.

Een veel gebruiken firmware voor 3D-printers is Marlin en dat is ook waar voor is gekozen voor de 3D-printer. Marlin is een open source firmware voor de RepRap-familie van 3D-printers. Het is afgeleid van Sprinter en grbl en werd een op zichzelf staand open source project op 12 augustus 2011 met de Github-release. Marlin heeft een licentie onder GPLv3 en is gratis voor alle toepassingen [3].

COMPILER

Een compiler word gebruikt om de firmware van de 3D-printer die in C++ is geschreven, om te zetten naar machinetaal. Machinetaal kan op de microprocessor worden uitgevoerd. Voor het compileren van Marlin is *Auto Build Marlin* een goede optie [4].

2 WAT ZIJN HDPE EN PP

HDPE is een hoge dichtheid variant van PE. PE is een thermoplastisch polymer dat in een breed scala aan toepassingen wordt gebruikt. HDPE is dus ook een thermoplast dat kan worden geëxtrudeerd.

PP is tegenwoordig een van de meest gebruikte plastics. Het is een polymer dat voornamelijk wordt gebruikt voor verpakkingen. HDPE is een thermoplast dat kan worden geëxtrudeerd. Het wordt geproduceerd via ketengroeipolymerisatie uit het monomeer propyleen.

2.1 WAAROM HDPE EN PP

HDPE en PP hebben een aantal gunstige eigenschappen die ervoor zorgen dat het makkelijk te bewerken is. Daarom is de meerderheid van de gebruikte plastics voor verpakkings-materiaal gemaakt van HDPE en PP. Omdat een grote hoeveelheid van de afvalstroom daarom ook uit HDPE en PP bestaan is het gunstig om dat te kunnen recyclen om mee te 3D-printen.

Inleiding

3 BESTANDEN DOWNLOADEN

Dit hele project is gewerkt in en is vastgelegd in git. Maar omdat de inhoud van het project en verslag intellectueel eigendom is van 3devo, staat dat openbaar op GitHub. Voor vragen over de inhoud van het project kun je contact opnemen met lucavanstraaten@icloud.com

1 ANALYSE VAN HET PROBLEEM

Waar de noodzaak van dit project word blootgelegd.

Kortweg is het probleem dat er momenteel niet met HDPE en PP kan worden ge-3D-print. 3D-printen met HDPE en PP is wenselijk omdat deze materialen overvloedig zijn in de huidige afvalstroom, en ze dus goedkoop te verkrijgen zijn.

Een voorbeeld van hoogwaardig PP in de afvalstroom zijn verpakkingen van medische goederen in ziekenhuizen. 3devo heeft PP bakjes van een oogheelkunde kliniek en wil dat recyclen tot 3D-printer filament. Zie Figuur 1.1 voor de bakjes.



Figuur 1.1: PP bakjes van een oogheelkunde kliniek

1.1 WAAROM IS EEN SPECIALE 3D-PRINTER NODIG

Er zijn twee redenen gegeven waarom een speciale printer nodig is voor het printen met HDPE en PP. Tests zijn uitgevoerd om vast te stellen of deze complicaties werkelijk het printen van HDPE en PP onmogelijk maken.

1.1.1 PRINTEN OP EEN CONVENTIONELE 3D-PRINTER

Er is een test uitgevoerd met het printen van HDPE en PP. Deze test zijn uitgevoerd op een conventionele FDM 3D-printer, namelijk een Ender-3. De slicer settings die gebruikt zijn waren vooral getweakt op temperatuur en snelheid.

1 Analyse van het probleem



Figuur 1.2: Test print met HDPE op een conventionele printer



Figuur 1.3: Test print met PP op een conventionele printer

LAAG HECHTING

De hypothese is dat bij het printen met PP de laag hechting niet goed zal zijn. Dat komt doordat PP onder de kristallisatieterminatuur niet "plakkerig" is. De test met het printen van PP heeft geconcludeerd dat die hypothese waar is. Het materiaal blijft moeilijk plakken aan het bed, en als dat goed ging, blijft het ook nauwelijks plakken aan zichzelf. Zie Figuur 1.3. Een ander probleem met het printen van PP is dat het flexibele plastic niet goed door een bowden extruder gaat vanwege de extra weerstand in dat systeem. Uitleg over wat een bowden tube extruder is staat in hoofdstuk 4.1.1.

KROM TREKKEN

De verwachting was dat het grote probleem met HDPE is dat het krom trekt tijdens het printen. Tijdens het printen zal er een warmte gradiënt ontstaan, omdat de laag die net is neergelegd al aan het afkoelen is voor dat de volgende de volgende laag daarop wordt gelegd. Tijdens het testen op een conventionele printer is vastgesteld dat dat inderdaad een probleem is, zie Figuur 1.2 voor een foto van de krom getrokken prints.

2 EISEN VAN HET PROJECT

2.0.1 RANDVOORWAARDEN

Het frame van een ender-5, de motoren van de ender-5, de behuizing, de warmte-elementen en de positie daar van waren allemaal al ontworpen voordat aan het project was begonnen. Dus deze aspecten van het project konden in principe niet veranderen.

2.0.2 FUNCTIONELE WENSEN

De verwarmde kamer van de printer moet maximaal 160 °C kunnen worden en stabiel zijn bij elke ingestelde temperatuur

2.0.3 GEBRUIKERSWENSEN

De gebruikersinterface moet een eenvoudige en snelle manier zijn om de printer te gebruiken, dat betekent in principe dat de printer op dezelfde manier te bedienen moet zijn als de originele Ender-5

2.0.4 ONTWERPBEPERKINGEN

Het project moet worden gedaan in 10 weken. Dus er kunnen geen onderdelen worden gebruikt met een lange levertijd.

De extruder en hotend kunnen niet op de conventionele manier worden gekoeld omdat dezen zich in de verwarmde kamer bevinden. Dus als blijkt dat dezen gekoeld moeten worden moet daar een andere oplossing voor worden gevonden.

3 PROBLEMEN

Tijdens het project zijn eer paar problemen opgetreden. Dezen hadden allemaal te maken met de extruder en dat daar geen plastic meer uit komt (vast liep).

3.1 SMELTEND PLASTIC IN DE EXTRUDER

Een probleem waar al vrij snel tegen aan werd gelopen is dat de extruder vast liep door dat het plastic te vroeg smelten, met het gevolg er geen plastic meer uit de nozel kwam. Dit gebeurde telkens rond de zelfde tijd na het starten van een print.

3.1.1 OORZAAK

Dit probleem was het resultaat van warme drive gears (extruder tandwielen). Heat creep is een term voor het warmte gradiënt door de metalen onderdelen van de extruder. De stappenmotor van de extruder en de heater cartridge van het hotend produceren allebei warmte. Deze warmte geleid door de hele extruder assemblage en komt dus ook bij de drive gears. Als daardoor het plastic ook warm word smelt het in de extruder. En kan het dus niet meer door de nozel worden geperst.

3.2 DUBBELVOUWEND PLASTIC IN DE EXTRUDER

Een probleem tijdens het testen met een PP print wat dat het plastic dubbel vouwde in de ruimte tussen de extruder tandwielen en de glijder naar de extruder. Dit is dus een ander probleem dan beschreven in hoofdstuk [3.1](#), dit probleem treed eerder op en zelfs als de kamer niet is verwarmd. Dit probleem treden ook op tijdens het testen van PP op een normale printer, echter om een andere reden.

3.2.1 OORZAAK

Dit komt doordat er genoeg ruimte is tussen de extruder tandwielen en het extruder frame dat er filament tussen door past. En omdat PP veel flexibeler is dan HDPE, gaat het makkelijk daar tussen zitten.

Dit probleem had een adere oorzaak bij het testen op een standaard ender-3. De oorzaak was echter dat de extruder veel meer kracht zou moeten zetten omdat de ender-3 een bowden extruder printer is. Uitleg over wat een bowden tube extruder is staat in hoofdstuk [4.1.1](#).

3.3 KAMER TEMPERATUUR OVERSHOOT

De temperatuur van de verwarmde kamer schiet door het setpoint heen voordat het stabiliseert of oscilleert rond de ingestelde waarden

3.3.1 OORZAAK

De massa van de thermistor is zo hoog dat het lang duurt om in evenwicht te komen met de luch-temperatuur. Daarom loopt de gemeente temperatuur dus aanzienlijk achter op de werkelijke temperatuur. De werkelijke temperatuur is geverifieerd door een thermokoppel naast de thermis-tor te hangen.

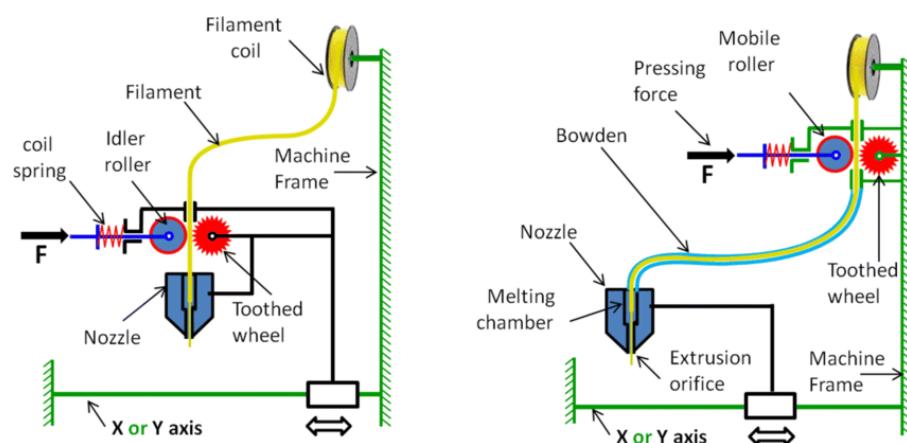
4

MOGELIJKE OPLOSSINGEN

4.1 EXTRUDER PROBLEEMEN

Het probleem van een vastlopende extruder kwam voor met een direct drive extruder. Een oplossing die overwogen was, was het overstappen naar een andere extruder architectuur.

4.1.1 VERSCHILLENDEN SOORTEN EXTRUDERS



Figuur 4.1: Diagram van de twee soorten extruders die veel woorden gebruikt [1].

BOWDEN EXTRUDER

De rechter helft van Figuur 4.1 [1] is een diagram van een Bowden extruder. Hierbij is te zien dat extruder los is van de hotend.

DIRECT DRIVE EXTRUDER

De linker helft van Figuur 4.1 [1] is een diagram van een direct drive extruder.

4.1.2 ZOU EEN BOWDEN EXTRUDER DE PROBLEMEN OPLOSSEN

Met een bowden extruder zou het probleem van smeltend plastic in de extruder (Hoofdstuk 3.1) opgelost kunnen worden. Echter was opgemerkt dat PP niet geprint kan worden met een bowden printer (Hoofdstuk 3.2).

4 Mogelijke oplossingen

4.1.3 ZOU EEN WATER GEKOELDE EXTRUDER DE PROBLEMEN OPLOSSEN

Een water gekoelde extruder/hotend zou allebei de problemen kunnen oplossen met als enige nadeel extra complexheid in de vorm van aanstuur elektronica en software voor de waterkoeling en de waterkoeling zelf (buizen, pomp, radiator).

Een goede optie zou de "Titan Aqua"[\[5\]](#) zijn.

4.2 TEMPERATUUR OVERSHOOT

Een PID lus voor de kamertemperatuur kan worden geïmplementeerd in de firmware van de printer, het vermogen van de warmte-elementen kan worden teruggedraaid om ervoor te zorgen dat de temperatuur minder snel stijgt en er dus een minder groot verschil is tussen de gemeenten en de werkelijke temperatuur.

5 DE GEKOZEN OPLOSSING

Om een oplossing te kiezen is overlegd met het materiaal team, en de opdrachtgever.

5.1 EXTRUDER

Voor de extruder is gekozen om de originele extruder er op te laten zitten, hier voor zijn 2 redenen:

- Er waren geen watergekoelde extruders beschikbaar met een levertijd voor het eind van de stageperiode, echter wordt wel aanbevolen in de toekomst om een water gekoelde extruder te installeren. Dat is ook opgenomen in hoofdstuk 9.
- Tijdens testen is geconcludeerd dat met bepaalde instelling HDPE prima printbaar is. En zeker als een paar massa procent carbon fiber wordt toegevoegd.

5.2 TEMPERATUUR OVERSHOOT

Een PID lus is geïmplementeerd in de firmware, en de autotune feature van Marlin is gebruikt om dezen te tunen.

6

ONTWERP VAN DE OPLOSSING EN DE BENODIGDE ONDERDELEN

6.1 HARDWARE

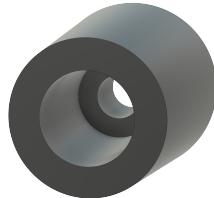
Het mechanisch ontwerp van de printer was grotendeels al gedaan, echter zijn er een aantal aanpassingen gedaan, die staan hier beschreven.

6.1.1 GE-3D-PRINTE OPLOSSINGEN

Een aantal problemen zijn opgelost door kleine onderdelen te 3D-printen. hier zijn daar een paar voorbeelden van.

VOETJES VAN DE PRINTER

De voetjes van de printer waren te kort, dus daar zijn langere voor ontworpen en ge-3D-print. Zie Figuur [6.1](#) voor een render van het 3D ontwerp.



Figuur 6.1: Render van het 3D ontwerp van de voetjes van de printer

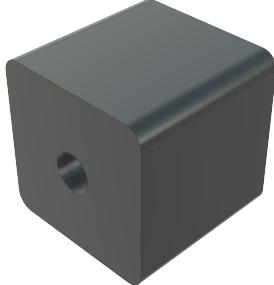
AFSTANDHOUDER

De originele printer is omgebouwd met roestvrijstalen panelen aan alle kanten. Om ervoor te zorgen dat er goede thermische isolatie is van de print kamer, is het een dubbelwandig ontwerp met glaswol er tussen. De dubbele wanden worden op afstand gehouden met ge-3D-prinete afstandhouders. Zie Figuur [6.2](#) voor een render van het 3D ontwerp van deze afstandhouders.

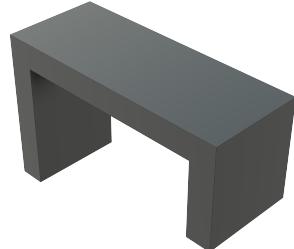
6 Ontwerp van de oplossing en de benodigde onderdelen

HAAKJE

Om de deur dicht te houden is er een haakje geprint. Zie Figuur [6.3](#) voor een render van het 3D ontwerp van het haakje.



Figuur 6.2: Render van het 3D ontwerp van de afstandhouder van de printer



Figuur 6.3: Render van het 3D ontwerp van het haakje van de printer

DIN-RAIL BEVESTIGINGEN

Op de zijkant van de printer was een DIN-rail bevestigd, hierop werd vervolgens alle elektronica bevestigd. In Figuur [6.4](#), [6.5](#) en [6.6](#) zijn de montage stukken te zien voor de PCB en de stroomtoevoer.



Figuur 6.4: Render van het 3D ontwerp van de DIN-rail PCB houder

6.2 ELECTRONICA

De te gebruiken elektronica was grotendeels al vastgesteld voor het project begon. Maar er was vrije keuze voor het hoofd board, hiervoor is een bigtreetech SKR-2 gekozen [\[6\]](#). Dit board was gekozen omdat deze vrij nieuw is, makkelijk te gebruiken, en veel mogelijkheden bied.



Figuur 6.5: Render van het 3D ontwerp van de schakelaarhouder op de DIN-rail



Figuur 6.6: Render van het 3D ontwerp van het haakje van de C-14 houder op de DIN-rail

6.3 SOFTWARE

Onder software word de slicer gezien. Er is gekozen voor prusaslicer, omdat dit een veel gebruikte slicer is die veel opties biedt en gratis(open source) is.

De settings waarmee is getest zijn beschikbaar gesteld voor het team bij 3devo. Zodat zij verder kunnen werken aan de progressie die tijdens de stageperiode is verricht. Dit staat gedocumenteerd in de GitHub repo van het project.

6.4 FIRMWARE

Voor de firmware is Marlin gekozen, en deze is geconfigureerd om te werken met de hardware (elektronica/stappenmotoren/sensoren) van de printer. Dit staat gedocumenteerd in de GitHub repo van het project.

7

ASSEMBLAGE

7.1 ASSEMBLAGE VAN DE PRINTER

Het mechanisch ontwerp van de printer was al vastgesteld, de printer hoefde dus alleen nog in elkaar woorden gezet. De printer is gebaseerd op de Ender-5, een robuuste printer die een goed platform biedt om een geïsoleerde behuizing er omheen te bouwen. De printer is voorzien van dubbele laser gesnelde wanden met glas wol daar tussen. Ondank dat de assemblage van de behuizing niet in de scoop van het project valt, was het wel nodig om verder te gaan met het project. dus dat was als eerste gedaan.

7.1.1 ELECTRONIC

Electronisch gezien moest er voral een hoop cable mangiment gebeuren. En de firmware moest op het bord worden gezet.

7.2 FIRMWARE

De firmware die op het bord draait is een aangepaste versie van Marlin. Marlin is een open source 3D-printer firmware die uitermate geschikt is voor hobby FDM 3D-printers, vanwege de wereldwijde support van vrijwilligers die meehelpen aan het project [3]. De aanpassingen aan de firmware zijn gemaakt in het configuratiebestand, zie bijlagen H en I voor de configuratie.

8

TOETSING EINDRESULTAAT AAN DE HAND VAN DE EISEN

8.1 VEILIGHEID

Er is rekening gehouden met de elektrische veiligheid, in figuur [6.6](#) is bijvoorbeeld te zien hoe een extra kapje is toegevoegd over de blote contacten van de C-14 stekker.

8.2 EENVOUDIGHEID

De printer is makkelijk te gebruiken aan de hande van de handlijding, zie bijlagen [G](#).

8.3 OVERIG

De printer voldoet ook aan de andere eisen.

9

CONCLUSIE EN AANBEVELINGEN

Uit dit verslag kan geconcludeerd worden dat dit project geslaagd is omdat het product op tijd werd opgeleverd, en voldoet aan alle gestelden eisen.

Echter zijn er wel wat aandachtpunten, zoals dat de printtijd niet volledig naar wens is, daarvoor is de aanbeveling aan 3devo om:

- Een gewater koelde extruder te installeren
- De printer beter te tunen om met gewenste kwaliteit te printen.

BIBLIOGRAFIE

- [1] M. Attaran, “Towards design of mechanical part and electronic control of multi-material/multicolor fused deposition modeling 3d printing,” *The International Journal of Advanced Manufacturing Technology*, vol. 110, 2020.
- [2] ——, “The rise of 3-d printing: The advantages of additive manufacturing over traditional manufacturing,” *Business Horizons*, vol. 60, no. 5, pp. 677–688, 2017. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0007681317300897>
- [3] S. Lahteine. (2021) What is Marlin? about marlin. [Online]. Available: marlinfw.org/docs/basics/introduction.html
- [4] ——. (2021) Auto build marlin. [Online]. Available: marlinfw.org/docs/basics/auto_build_marlin.html
- [5] E3D. (2021) Titan aqua. [Online]. Available: e3d-online.com/products/titan-aqua
- [6] bigtreetech. (2021) Skr-2. [Online]. Available: github.com/bigtreetech/SKR-2

Appendices

A INTERVISIE

Student over Intervisie bedrijf			
Naam bedrijf		EFPC	
Vestigingsplaats		De Bilt	
Bedrijfsbegeleider		Ronald van brakel	
Naam student		Menno Hondius	
Datum		11-10-21	
Type bedrijfservaring	<input type="radio"/> eerste stage	<input checked="" type="radio"/> tweede stage	<input type="radio"/> afstuderen

Omschrijving van de verrichte werkzaamheden:

Inzicht verkrijgen in brand risico als gevolg van zonen installaties op fabriek daken. De installatie van zonnepanelen heeft een bepaald risicoprofiel waarbij elektriciteit een bepaalde rol heeft.

Omschrijving van de werkomgeving en/of de werksfeer:

Informeel, begeleider makkelijk aan te spreken. Altijd vragen om hulp en veel mensen met kennis om vragen aan te stellen.

Algemene indruk en/of bijzonderheden van de stage:

Er is een cursus ruimte waar sprinkler cursussen worden georganiseerd.

B

STUDENT OVER DOCENTBEZOEK

Formulier HHS Elektrotechniek

Student over docentbezoek		
Naam bedrijf		3devo
Vestigingsplaats		Utrecht
Bedrijfsbegeleider	Jesse	
Naam student	Luca van Straaten	
Naam docent	Ad van der Bergh	
Datum	29-9-21	
Type bedrijfservaring	X eerste stage	<input type="radio"/> tweede stage <input type="radio"/> afstuderen

Besproken onderwerpen:

Wat 3devo doet als bedrijf?

Wat ik doe bij 3devo?

Waarom ik dat doe bij 3devo?

Wat de moeilijkheden zijn met mijn opdracht?

Aandachtspunten m.b.t. mijn p.v.a.

Gemaakte afspraken:

Ik ga mijn p.v.a. verbeteren en opsturen.

Conclusie van het bezoek:

Ik ben goed op weg en als er vragen zijn kan ik die altijd op de mail zetten.

C BEDRIJF OVER STUDENT

Formulier stagebeoordeling Elektrotechniek

Naam Student	Luca van Straaten
Studentnummer	18073611
Stage 1 / 2	1
Shift 1 / 2 / 3 / 4	1
Bedrijf	3devo
Bedrijfsbegeleider	Jesse van der Zouw

Beoordeling door het bedrijf

	O	T	V	G	U
Inzicht in opdracht				x	
Nauwkeurigheid				x	
Inzet			x		
Initiatief			x		
Houding als werknemer			x		
Samenwerking met begeleider			x		
Samenwerking met collega's			x		
	O	T	V	G	U
Competentie Analyseren ²⁾			x		
Competentie Ontwerpen ²⁾			x		
Competentie Realiseren ¹⁾				x	
Competentie Managen ²⁾			x		
Competentie Onderzoeken ²⁾			x		
Competentie Professionaliseren ¹⁾			x		

	O	T	V	G	U
Totale beoordeling			x		

Student (voor gezien) datum

Luca van Straaten	15/11/2021
--------------------------	-------------------

Namens het bedrijf datum

Jesse van der Zouw	15/11/2021
---------------------------	-------------------

O = onvoldoende; T = twijfelachtig; V = voldoende; G = goed; U = uitmuntend; NVT = Niet van toepassing

¹⁾ In de eerste stage van de student

²⁾ In de tweede stage van de student

D STUDENT OVER BEDRIJF

Formulier HHS Elektrotechniek

Student over bedrijf	
Naam bedrijf	3Devo
Vestigingsplaats	utrecht
Bedrijfsbegeleider	Jesse van der Zouw
Naam student	Luca van Straaten
Datum	12-11-21
Type bedrijfservaring	<input checked="" type="checkbox"/> eerste stage <input type="radio"/> tweede stage <input type="radio"/> afstuderen

Omschrijving van de verrichtte werkzaamheden:

Werken aan prototypen 3D-printer

Omschrijving van de werkomgeving en/of de werksfeer:

De werkomgeving bij 3devo is relaxed, er wordt van je verwacht dat je goed werk levert, maar er wordt absoluut niet gemicromanaged

Algemene indruk en/of bijzonderheden van de stage:

Ik heb nou samengewerkt met het materiaal team, en minder met mijn begeleider die elektronica ontwerp doet

E VERANTWOORDING VAN DE BEHAALDE COMPETENTIES

verantwoording van de behaalde competenties

Realiseren

- Ik heb de printer opgeleverd die voldoet aan de gestelde eisen.
- IK heb daarvoor de printer in elkaar moeten zetten, en sommige onderdelen zelf moeten produceren.
- Ik heb de printer getest met meerdere materialen.
- Ik heb een verslag opgeleverd met daarin documentatie van het realisatieproces.

Professionaliseren

- Ik ben zelfstandig te werk gegaan met het project.
- Ik ben flexibel geweest op kantoor, door een aantal extra taken op me te nemen.
- Ik heb al mijn beslissingen genomen met de ethiek daar achter in gedachten.
- Ik heb mijn collega's geholpen en geadviseerd met probleem waar zij tegen aan liepen.
- Ik heb de software voor interen communicatie binnen het bedrijf gebruikt, en daar zowel in het Nederlands als in het Engels op gecommuniceerd.

Ontwerpen

- Ik heb de elektronica gekozen die voldoet aan de gestelde eisen.
- In het verslag is het ontwerp beschreven.
- Het ontwerp is getest aan de hand van de eisen.
- Ik heb documentatie opgeleverd waarmee de opdrachtgever de printer kan gebruiken.

F

KORTE BESCHRIJVING VAN DE
BEDRIJFSORGANISATIE

korte beschrijving van de bedrijfsorganisatie

Het bedrijf zit heel gewoontjes in elkaar. er is een baas, hij stuurt een aantal medewerkers aan, die vervolgens ook een aantal medewerkers aansturen. Maar het is ook heel open, iedereen kan aan iedereen om hulp vragen, en dat gebeurt ook regelmatig.

G

OVERDRACHT DOCUMENT

Transfer document for the heated chamber 3D printer (solid-kinetic)

This document describes how the printer can used.

Parts of the printer

- main bord: BigTreeTech SKR 2
- Hotend: E3D Titan Aero 1,75 mm 24V
- Printer platform: Creality 3D Ender-5
- Volume (X, Y, Z): 200x200x300 mm

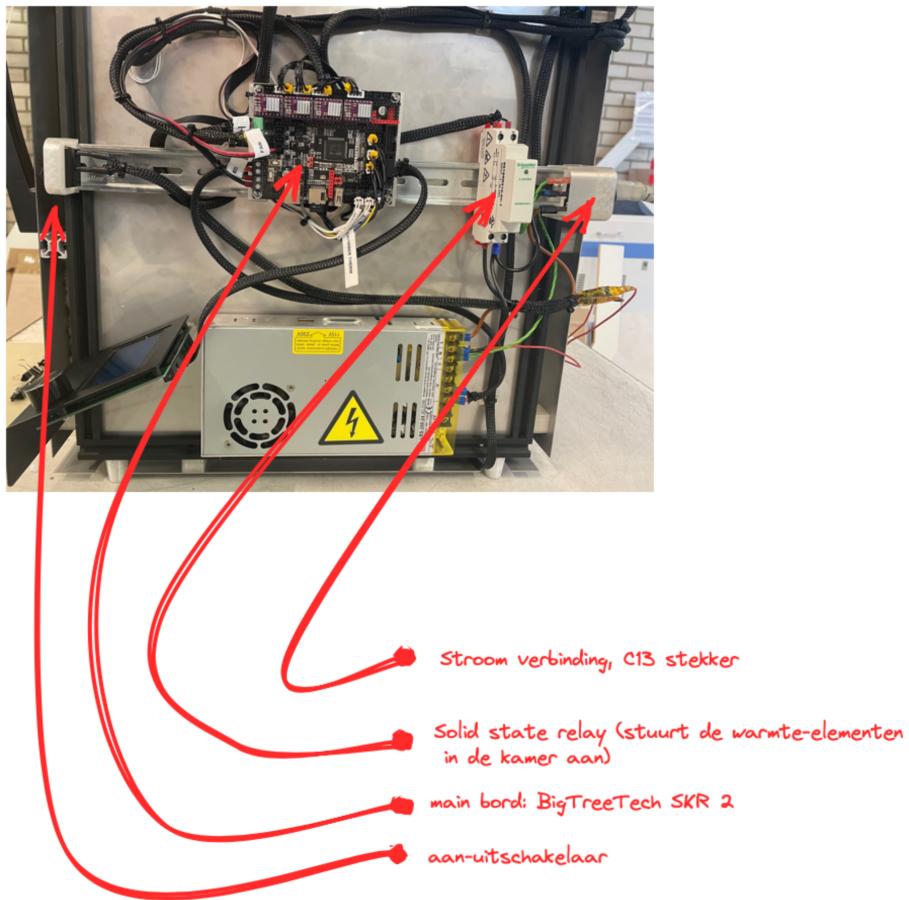


Figure 1: components

What do you have to do to use the printer?

1. Turn on the printer

Is the printer plugged in and the power switch turned on?

2. Slice:

Prusa slicer (or other Slic3r forks)

install: Install the settings files found on GitHub : <https://github.com/lucanatorvs/solid-kinetic> the settings files are in the folder: `slicer_settings/PrusaSlicer` and there are also the installation instructions.

using the slicer: Instructions for the slicer are on the prusa3d website.
<https://www.prusa3d.com/prusaslicer/>

setting up the slicer: The slicer has a number of settings that you can adjust. that is necessary to get good results. But the project defaults on github are good enough to start with.

To completely tune the printer and achieve the best results, you can use the tool by Teaching Tech <https://teachingtechyt.github.io/calibration.html>. Here is clearly documented what you can do to calibrate the printer.

export gcode: From the slicer, you can export the gcode to a micro-SD card. The printer prints directly from the SD card. If you have trouble connecting to the card, try turning the printer off and on, or try reformatting the SD-card to FAT32.

3. printing:

Print from the SD-card. Most print settings can be directly changed during the print from the display.

4. Bed (leveling):

The bed leveling procedure is the same as any other 3D printer. Or use the mesh leveling tool. The bed is covered with PP tape, this can be scraped off and replaced when needed.

5. Filament:

The printer uses 1.75 mm filament. The spool needs to be supported above the printer and the filament needs to be fed through the hole on top of the printer into the E3D Titan Aero extruder-hotend.

6 heated chamber:

The heated chamber can either be controlled from G-CODE or from the printer directly. The G-CODE commands are:

```
M141 S40 ; set chamber temp @ 40  
M191 S40 ; wait for chamber temp @ 40
```

furure improvements

Add a water cooled hotend-extruder.



Figure 2: qrcode to the github repo

H CONFIGURATION

```
1 /**
2  * Marlin 3D Printer Firmware
3  * Copyright (c) 2020 MarlinFirmware
4  * [https://github.com/MarlinFirmware/Marlin]
5  *
6  * Based on Sprinter and grbl.
7  * Copyright (c) 2011 Camiel Gubbels / Erik van der Zalm
8  *
9  * This program is free software: you can redistribute it and/or modify
10 * it under the terms of the GNU General Public License as published by
11 * the Free Software Foundation, either version 3 of the License, or
12 * (at your option) any later version.
13 *
14 * This program is distributed in the hope that it will be useful,
15 * but WITHOUT ANY WARRANTY; without even the implied warranty of
16 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
17 * GNU General Public License for more details.
18 *
19 * You should have received a copy of the GNU General Public License
20 * along with this program. If not, see <https://www.gnu.org/licenses/>.
21 */
22 #pragma once
23
24 /**
25  * Configuration.h
26  *
27  * Basic settings such as:
28  *
29  * - Type of electronics
30  * - Type of temperature sensor
31  * - Printer geometry
32  * - Endstop configuration
33  * - LCD controller
34  * - Extra features
35  *
36  * Advanced settings can be found in Configuration_adv.h
37 */
38 #define CONFIGURATION_H_VERSION 02000902
39
40 //=====
41 =
42 //===== Getting Started
43 =
44 /**
45  * Here are some useful links to help get your machine configured and
46  * calibrated:
47  *
48  * Example Configs:
49  * https://github.com/MarlinFirmware/Configurations/branches/all
50  *
51  * Průša Calculator: https://blog.prusaprinters.org/calculator_3416/
52  *
53  * Calibration Guides: https://reprap.org/wiki/Calibration
54 *
```

```
https://reprap.org/wiki/Tritton_Hunter%27s_Calibration_Guide
53 *
54 https://sites.google.com/site/repraplogphase/calibration-of-your-reprap
55 *
56 * Calibration Objects: https://www.thingiverse.com/thing:5573
57 * https://www.thingiverse.com/thing:1278865
58 */
59
60 //=====
61 //=====
62 //===== DELTA / SCARA / TPARA
63 //=====
64 // Download configurations from the link above and customize for your
65 // machine.
66 // Examples are located in config/examples/delta, .../SCARA, and .../TPARA.
67 //
68 //=====
69 // @section info
70
71 // Author info of this build printed to the host during boot and M115
72 #define STRING_CONFIG_H_AUTHOR "(Lucanator, Io)" // Who made the changes.
73 // #define CUSTOM_VERSION_FILE Version.h // Path from the root directory (no
74 // quotes)
75 /**
76 * *** VENDORS PLEASE READ ***
77 *
78 * Marlin allows you to add a custom boot image for Graphical LCDs.
79 * With this option Marlin will first show your custom screen followed
80 * by the standard Marlin logo with version number and web URL.
81 *
82 * We encourage you to take advantage of this new feature and we also
83 * respectfully request that you retain the unmodified Marlin boot screen.
84 */
85
86 // Show the Marlin bootscreen on startup. ** ENABLE FOR PRODUCTION **
87 // #define SHOW_BOOTSCREEN
88
89 // Show the bitmap in Marlin/_Bootscreen.h on startup.
90 // #define SHOW_CUSTOM_BOOTSCREEN
91
92 // Show the bitmap in Marlin/_Statusscreen.h on the status screen.
93 // #define CUSTOM_STATUS_SCREEN_IMAGE
94
95 // @section machine
96
97 /**
98 * Select the serial port on the board to use for communication with the
99 * host.
100 * This allows the connection of wireless adapters (for instance) to non-
101 * default port pins.
102 * Serial port -1 is the USB emulated serial port, if available.
103 * Note: The first serial port (-1 or 0) will always be used by the Arduino
```

```
-----  
bootloader.  
102  
103 * :[-1, 0, 1, 2, 3, 4, 5, 6, 7]  
104 */  
105#define SERIAL_PORT 1  
106  
107/**  
108 * Serial Port Baud Rate  
109 * This is the default communication speed for all serial ports.  
110 * Set the baud rate defaults for additional serial ports below.  
111 *  
112 * 250000 works in most cases, but you might try a lower speed if  
113 * you commonly experience drop-outs during host printing.  
114 * You may try up to 1000000 to speed up SD file transfer.  
115 *  
116 * :[2400, 9600, 19200, 38400, 57600, 115200, 250000, 500000, 1000000]  
117 */  
118#define BAUDRATE 115200  
119//#define BAUD_RATE_GCODE      // Enable G-code M575 to set the baud rate  
120  
121/**  
122 * Select a secondary serial port on the board to use for communication with  
the host.  
123 * Currently Ethernet (-2) is only supported on Teensy 4.1 boards.  
124 * :[-2, -1, 0, 1, 2, 3, 4, 5, 6, 7]  
125 */  
126#define SERIAL_PORT_2 -1  
127//#define BAUDRATE_2 250000    // Enable to override BAUDRATE  
128  
129/**  
130 * Select a third serial port on the board to use for communication with the  
host.  
131 * Currently only supported for AVR, DUE, LPC1768/9 and STM32/STM32F1  
132 * :[-1, 0, 1, 2, 3, 4, 5, 6, 7]  
133 */  
134#define SERIAL_PORT_3 3  
135//#define BAUDRATE_3 250000    // Enable to override BAUDRATE  
136  
137// Enable the Bluetooth serial interface on AT90USB devices  
138//#define BLUETOOTH  
139  
140// Choose the name from boards.h that matches your setup  
141#ifndef MOTHERBOARD  
142    #define MOTHERBOARD BOARD_BTT_SKR_V2_0_REV_B  
143#endif  
144  
145// Name displayed in the LCD "Ready" message and Info menu  
146#define CUSTOM_MACHINE_NAME "3Devo"  
147  
148// Printer's unique ID, used by some programs to differentiate between  
machines.  
149// Choose your own or use a service like  
https://www.uuidgenerator.net/version4  
150//#define MACHINE_UUID "f1448366-6e98-4b6f-bced-90fb6e24768f"  
151  
152/**  
153 * Define the number of coordinated linear axes.  
* See https://github.com/DerAndere1/Marlin/wiki
```

```

155 * Each linear axis gets its own stepper control and endstop:
156 *
157 * Steppers: *_STEP_PIN, *_ENABLE_PIN, *_DIR_PIN, *_ENABLE_ON
158 * Endstops: *_STOP_PIN, USE_*MIN_PLUG, USE_*MAX_PLUG
159 * Axes: *_MIN_POS, *_MAX_POS, INVERT_*_DIR
160 * Planner: DEFAULT_AXIS_STEPS_PER_UNIT, DEFAULT_MAX_FEEDRATE
161 * DEFAULT_MAX_ACCELERATION, AXIS_RELATIVE_MODES,
162 * MICROSTEP_MODES, MANUAL_FEEDRATE
163 *
164 * :[3, 4, 5, 6]
165 */
166 // #define LINEAR_AXES 3
167
168 /**
169 * Axis codes for additional axes:
170 * This defines the axis code that is used in G-code commands to
171 * reference a specific axis.
172 * 'A' for rotational axis parallel to X
173 * 'B' for rotational axis parallel to Y
174 * 'C' for rotational axis parallel to Z
175 * 'U' for secondary linear axis parallel to X
176 * 'V' for secondary linear axis parallel to Y
177 * 'W' for secondary linear axis parallel to Z
178 * Regardless of the settings, firmware-internal axis IDs are
179 * I (AXIS4), J (AXIS5), K (AXIS6).
180 */
181 #if LINEAR_AXES >= 4
182     #define AXIS4_NAME 'A' // :['A', 'B', 'C', 'U', 'V', 'W']
183 #endif
184 #if LINEAR_AXES >= 5
185     #define AXIS5_NAME 'B' // :['A', 'B', 'C', 'U', 'V', 'W']
186 #endif
187 #if LINEAR_AXES >= 6
188     #define AXIS6_NAME 'C' // :['A', 'B', 'C', 'U', 'V', 'W']
189 #endif
190
191 // @section extruder
192
193 // This defines the number of extruders
194 // :[0, 1, 2, 3, 4, 5, 6, 7, 8]
195 #define EXTRUDERS 1
196
197 // Generally expected filament diameter (1.75, 2.85, 3.0, ...). Used for
198 // Volumetric, Filament Width Sensor, etc.
199 #define DEFAULT_NOMINAL_FILAMENT_DIA 1.75
200
201 // For Cyclops or any "multi-extruder" that shares a single nozzle.
202 // #define SINGLENOZZLE
203
204 // Save and restore temperature and fan speed on tool-change.
205 // Set standby for the unselected tool with M104/106/109 T...
206 #if ENABLED(SINGLENOZZLE)
207     // #define SINGLENOZZLE_STANDBY_TEMP
208     // #define SINGLENOZZLE_STANDBY_FAN
209 #endif
210
211 /**
212 * Multi-Material Unit
213 * Set to one of these predefined models:

```

```

--> See the end of these pre-defined models.
213 *
214 * PRUSA_MMU1           : Průša MMU1 (The "multiplexer" version)
215 * PRUSA_MMU2           : Průša MMU2
216 * PRUSA_MMU2S          : Průša MMU2S (Requires MK3S extruder with motion
sensor, EXTRUDERS = 5)
217 * EXTENDABLE_EMU_MMU2  : MMU with configurable number of filaments (ERCF,
SMUFF or similar with Průša MMU2 compatible firmware)
218 * EXTENDABLE_EMU_MMU2S : MMUS with configurable number of filaments
(ERCF, SMUFF or similar with Průša MMU2 compatible firmware)
219 *
220 * Requires NOZZLE_PARK_FEATURE to park print head in case MMU unit fails.
221 * See additional options in Configuration_adv.h.
222 */
223 //#define MMU_MODEL PRUSA_MMU2
224
225 // A dual extruder that uses a single stepper motor
226 //#define SWITCHING_EXTRUDER
227 #if ENABLED(SWITCHING_EXTRUDER)
228   #define SWITCHING_EXTRUDER_SERVO_NR 0
229   #define SWITCHING_EXTRUDER_SERVO_ANGLES { 0, 90 } // Angles for E0, E1[,,
E2, E3]
230   #if EXTRUDERS > 3
231     #define SWITCHING_EXTRUDER_E23_SERVO_NR 1
232   #endif
233 #endif
234
235 // A dual-nozzle that uses a servomotor to raise/lower one (or both) of the
nozzles
236 //#define SWITCHING_NOZZLE
237 #if ENABLED(SWITCHING_NOZZLE)
238   #define SWITCHING_NOZZLE_SERVO_NR 0
239   // #define SWITCHING_NOZZLE_E1_SERVO_NR 1           // If two servos are
used, the index of the second
240   #define SWITCHING_NOZZLE_SERVO_ANGLES { 0, 90 } // Angles for E0, E1
(single servo) or lowered/raised (dual servo)
241 #endif
242
243 /**
244 * Two separate X-carriages with extruders that connect to a moving part
245 * via a solenoid docking mechanism. Requires SOL1_PIN and SOL2_PIN.
246 */
247 //#define PARKING_EXTRUDER
248
249 /**
250 * Two separate X-carriages with extruders that connect to a moving part
251 * via a magnetic docking mechanism using movements and no solenoid
252 *
253 * project  : https://www.thingiverse.com/thing:3080893
254 * movements : https://youtu.be/0xCEiG9VS3k
255 *             https://youtu.be/BqbcsoCU2FE
256 */
257 //#define MAGNETIC_PARKING_EXTRUDER
258
259 #if EITHER(PARKING_EXTRUDER, MAGNETIC_PARKING_EXTRUDER)
260
261   #define PARKING_EXTRUDER_PARKING_X { -78, 184 } // X positions for
parking the extruders
262   #define PARKING_EXTRUDER_GRAB_DISTANCE 1        // (mm) Distance to

```

```

move beyond the parking point to grab the extruder
263 // #define MANUALSOLENOID_CONTROL // Manual control of
docking solenoids with M380 S / M381
264
265 #if ENABLED(PARKING_EXTRUDER)
266
267     #define PARKING_EXTRUDER_SOLENOIDS_INVERT           // If enabled, the
solenoid is NOT magnetized with applied voltage
268     #define PARKING_EXTRUDER_SOLENOIDS_PINS_ACTIVE_LOW    // LOW or HIGH pin
signal energizes the coil
269     #define PARKING_EXTRUDER_SOLENOIDS_DELAY 250          // (ms) Delay for
magnetic field. No delay if 0 or not defined.
270     // #define MANUALSOLENOID_CONTROL // Manual control of
docking solenoids with M380 S / M381
271
272 #elif ENABLED(MAGNETIC_PARKING_EXTRUDER)
273
274     #define MPE_FAST_SPEED      9000        // (mm/min) Speed for travel
before last distance point
275     #define MPE_SLOW_SPEED      4500        // (mm/min) Speed for last
distance travel to park and couple
276     #define MPE_TRAVEL_DISTANCE 10          // (mm) Last distance point
277     #define MPE_COMPENSATION     0           // Offset Compensation -1 , 0 , 1
(multiplier) only for coupling
278
279 #endif
280
281#endif
282
283 /**
284 * Switching Toolhead
285 *
286 * Support for swappable and dockable toolheads, such as
287 * the E3D Tool Changer. Toolheads are locked with a servo.
288 */
289 // #define SWITCHING_TOOLHEAD
290
291 /**
292 * Magnetic Switching Toolhead
293 *
294 * Support swappable and dockable toolheads with a magnetic
295 * docking mechanism using movement and no servo.
296 */
297 // #define MAGNETIC_SWITCHING_TOOLHEAD
298
299 /**
300 * Electromagnetic Switching Toolhead
301 *
302 * Parking for CoreXY / HBot kinematics.
303 * Toolheads are parked at one edge and held with an electromagnet.
304 * Supports more than 2 Toolheads. See https://youtu.be/JolbsAKTKf4
305 */
306 // #define ELECTROMAGNETIC_SWITCHING_TOOLHEAD
307
308 #if ANY(SWITCHING_TOOLHEAD, MAGNETIC_SWITCHING_TOOLHEAD,
ELECTROMAGNETIC_SWITCHING_TOOLHEAD)
309     #define SWITCHING_TOOLHEAD_Y_POS      235        // (mm) Y position
of the toolhead dock
310     #define SWTTCHNG_TOOLHEAD_Y_SECURITY 10         // (mm) Security

```

```

300 // ACTUATING SWITCHING_TOOLHEAD_Y_SECURITY
301 #define SWITCHING_TOOLHEAD_Y_CLEAR 10 // (mm) Security
302 distance from dock for unobstructed X axis
303 #define SWITCHING_TOOLHEAD_X_POS 60 // (mm) Minimum
304 for parking the extruders { 215, 0 } // (mm) X positions
305 #if ENABLED(SWITCHING_TOOLHEAD)
306   #define SWITCHING_TOOLHEAD_SERVO_NR 2 // Index of the
307   servo connector
308   #define SWITCHING_TOOLHEAD_SERVO_ANGLES { 0, 180 } // (degrees) Angles
309   for Lock, Unlock
310   #elif ENABLED(MAGNETIC_SWITCHING_TOOLHEAD)
311     #define SWITCHING_TOOLHEAD_Y_RELEASE 5 // (mm) Security
312     distance Y axis
313     #define SWITCHING_TOOLHEAD_X_SECURITY { 90, 150 } // (mm) Security
314     distance X axis (T0,T1)
315     // #define PRIME_BEFORE_REMOVE // Prime the nozzle
316     before release from the dock
317     #if ENABLED(PRIME_BEFORE_REMOVE)
318       #define SWITCHING_TOOLHEAD_PRIME_MM 20 // (mm) Extruder
319       prime length
320       #define SWITCHING_TOOLHEAD_RETRACT_MM 10 // (mm) Retract
321       after priming length
322       #define SWITCHING_TOOLHEAD_PRIME_FEEDRATE 300 // (mm/min) Extruder
323       prime feedrate
324       #define SWITCHING_TOOLHEAD_RETRACT_FEEDRATE 2400 // (mm/min) Extruder
325       retract feedrate
326     #endif
327     #elif ENABLED(ELECTROMAGNETIC_SWITCHING_TOOLHEAD)
328       #define SWITCHING_TOOLHEAD_Z_HOP 2 // (mm) Z raise for
329       switching
330     #endif
331   #endif
332   #endif
333   #endif
334
335 /**
336  * "Mixing Extruder"
337  * - Adds G-codes M163 and M164 to set and "commit" the current mix
338  * factors.
339  * - Extends the stepping routines to move multiple steppers in proportion
340  * to the mix.
341  * - Optional support for Repetier Firmware's 'M164 S<index>' supporting
342  * virtual tools.
343  * - This implementation supports up to two mixing extruders.
344  * - Enable DIRECT_MIXING_IN_G1 for M165 and mixing in G1 (from Pia
345  * Taubert's reference implementation).
346  */
347 // #define MIXING_EXTRUDER
348 #if ENABLED(MIXING_EXTRUDER)
349   #define MIXING_STEPPERS 2 // Number of steppers in your mixing
350   extruder
351   #define MIXING_VIRTUAL_TOOLS 16 // Use the Virtual Tool method with M163
352   and M164
353   // #define DIRECT_MIXING_IN_G1 // Allow ABCDHI mix factors in G1
354   movement commands
355   // #define GRADIENT_MIX // Support for gradient mixing with M166
356   and LCD
357   // #define MIXING_PRESETS // Assign 8 default V-tool presets for 2
358   or 3 MIXING_STEPPERS
359   #if ENABLED(GRADIENT_MIX)

```

```

347     // #define GRADIENT_VTOOL           // Add M166 T to use a V-tool index as a
348     Gradient alias
349     #endif
350   #endif
351
351 // Offset of the extruders (uncomment if using more than one and relying on
352 // firmware to position when changing).
352 // The offset has to be X=0, Y=0 for the extruder 0 hotend (default
352 // extruder).
353 // For the other hotends it is their distance from the extruder 0 hotend.
354 // #define HOTEND_OFFSET_X { 0.0, 20.00 } // (mm) relative X-offset for each
354 // nozzle
355 // #define HOTEND_OFFSET_Y { 0.0, 5.00 } // (mm) relative Y-offset for each
355 // nozzle
356 // #define HOTEND_OFFSET_Z { 0.0, 0.00 } // (mm) relative Z-offset for each
356 // nozzle
357
358 // @section machine
359
360 /**
361 * Power Supply Control
362 *
363 * Enable and connect the power supply to the PS_ON_PIN.
364 * Specify whether the power supply is active HIGH or active LOW.
365 */
366 // #define PSU_CONTROL
367 // #define PSU_NAME "Power Supply"
368
369 #if ENABLED(PSU_CONTROL)
370     // #define MKS_PWC                  // Using the MKS PWC add-on
371     // #define PS_OFF_CONFIRM          // Confirm dialog when power off
372     // #define PS_OFF_SOUND            // Beep 1s when power off
373     #define PSU_ACTIVE_STATE LOW      // Set 'LOW' for ATX, 'HIGH' for X-Box
374
375     // #define PSU_DEFAULT_OFF        // Keep power off until enabled directly
375 // with M80
376     // #define PSU_POWERUP_DELAY 250    // (ms) Delay for the PSU to warm up to
376 // full power
377
378     // #define PSU_POWERUP_GCODE "M355 S1" // G-code to run after power-on
378 // (e.g., case light on)
379     // #define PSU_POWEROFF_GCODE "M355 S0" // G-code to run before power-off
379 // (e.g., case light off)
380
381     // #define AUTO_POWER_CONTROL      // Enable automatic control of the PS_ON
381 // pin
382     #if ENABLED(AUTO_POWER_CONTROL)
383         #define AUTO_POWER_FANS        // Turn on PSU if fans need power
384         #define AUTO_POWER_E_FANS
385         #define AUTO_POWER_CONTROLLERFAN
386         #define AUTO_POWER_CHAMBER_FAN
387         #define AUTO_POWER_COOLER_FAN
388         // #define AUTO_POWER_E_TEMP      50 // (°C) Turn on PSU if any
388 // extruder is over this temperature
389         // #define AUTO_POWER_CHAMBER_TEMP 30 // (°C) Turn on PSU if the chamber
389 // is over this temperature
390         // #define AUTO_POWER_COOLER_TEMP 26 // (°C) Turn on PSU if the cooler
390 // is over this temperature
391         #define DOWNGD_TTMFAULT        20 // (°C) Turn off power if the

```

```

391  * @section POWER_TIMEOUT
392  * machine is idle for this duration
393  * // #define POWER_OFF_DELAY           60 // (s) Delay of poweroff after M81
394  * command. Useful to let fans run for extra time.
395  * #endif
396  * #endif
397  * //=====
398  * //=====
399  * // @section temperature
400  */
401 /**
402  * --NORMAL IS 4.7kΩ PULLUP!-- 1kΩ pullup can be used on hotend sensor,
403  * using correct resistor and table
404  *
405  * Temperature sensors available:
406  *
407  * SPI RTD/Termocouple Boards - IMPORTANT: Read the NOTE below!
408  * -----
409  *      -5 : MAX31865 with Pt100/Pt1000, 2, 3, or 4-wire (only for sensors 0-
410  * 1)
411  *          NOTE: You must uncomment/set the MAX31865_*_OHMS_n
412  * defines below.
413  *      -3 : MAX31855 with Thermocouple, -200°C to +700°C (only for sensors 0-
414  * 1)
415  *      -2 : MAX6675 with Thermocouple, 0°C to +700°C (only for sensors 0-
416  * 1)
417  *
418  * NOTE: Ensure TEMP_n_CS_PIN is set in your pins file for each
419  * TEMP_SENSOR_n using an SPI Thermocouple. By default,
420  * Hardware SPI on the default serial bus is used. If you have also
421  * set TEMP_n_SCK_PIN and TEMP_n_MISO_PIN,
422  * Software SPI will be used on those ports instead. You can force
423  * Hardware SPI on the default bus in the
424  * Configuration_adv.h file. At this time, separate Hardware SPI
425  * buses for sensors are not supported.
426  *
427  * Analog Thermocouple Boards
428  * -----
429  *      -4 : AD8495 with Thermocouple
430  *      -1 : AD595 with Thermocouple
431  *
432  * Analog Thermistors - 4.7kΩ pullup - Normal
433  * -----
434  *      1 : 100kΩ EPCOS - Best choice for EPCOS thermistors
435  *      331 : 100kΩ Same as #1, but 3.3V scaled for MEGA
436  *      332 : 100kΩ Same as #1, but 3.3V scaled for DUE
437  *      2 : 200kΩ ATC Semitec 204GT-2
438  *      202 : 200kΩ Copymaster 3D
439  *      3 : ???Ω Mendel-parts thermistor
440  *      4 : 10kΩ Generic Thermistor !! DO NOT use for a hotend - it gives
441  * bad resolution at high temp. !!
442  *      5 : 100kΩ ATC Semitec 104GT-2/104NT-4-R025H42G - Used in ParCan, J-
443  * Head, and E3D, SliceEngineering 300°C
444  *      501 : 100kΩ Zonestar - Tronxy X3A

```

```
434 * 502 : 100kΩ Zonestar – used by hot bed in Zonestar Průša P802M
435 * 512 : 100kΩ RPW-Ultra hotend
436 * 6 : 100kΩ EPCOS – Not as accurate as table #1 (created using a fluke
thermocouple)
437 * 7 : 100kΩ Honeywell 135-104LAG-J01
438 * 71 : 100kΩ Honeywell 135-104LAF-J01
439 * 8 : 100kΩ Vishay 0603 SMD NTCS0603E3104FXT
440 * 9 : 100kΩ GE Sensing AL03006-58.2K-97-G1
441 * 10 : 100kΩ RS PRO 198-961
442 * 11 : 100kΩ Keenovo AC silicone mats, most Wanhao i3 machines – beta
3950, 1%
443 * 12 : 100kΩ Vishay 0603 SMD NTCS0603E3104FXT (#8) – calibrated for
Makibox hot bed
444 * 13 : 100kΩ Hisens up to 300°C – for "Simple ONE" & "All In ONE"
hotend – beta 3950, 1%
445 * 15 : 100kΩ Calibrated for JGAurora A5 hotend
446 * 18 : 200kΩ ATC Semitec 204GT-2 Dagoma.Fr – MKS_Base_DKU001327
447 * 22 : 100kΩ GTM32 Pro vB – hotend – 4.7kΩ pullup to 3.3V and 220Ω to
analog input
448 * 23 : 100kΩ GTM32 Pro vB – bed – 4.7kΩ pullup to 3.3v and 220Ω to
analog input
449 * 30 : 100kΩ Kis3d Silicone heating mat 200W/300W with 6mm precision
cast plate (EN AW 5083) NTC100K – beta 3950
450 * 60 : 100kΩ Maker's Tool Works Kapton Bed Thermistor – beta 3950
451 * 61 : 100kΩ Formbot/Vivedino 350°C Thermistor – beta 3950
452 * 66 : 4.7MΩ Dyze Design High Temperature Thermistor
453 * 67 : 500kΩ SliceEngineering 450°C Thermistor
454 * 70 : 100kΩ bq Hephestos 2
455 * 75 : 100kΩ Generic Silicon Heat Pad with NTC100K MGB18-104F39050L32
456 * 2000 : 100kΩ Ultimachine Rambo TDK NTCG104LH104KT1 NTC100K motherboard
Thermistor
457 *
458 * Analog Thermistors – 1kΩ pullup – Atypical, and requires changing out
the 4.7kΩ pullup for 1kΩ.
459 * ----- (but gives greater accuracy and more
stable PID)
460 * 51 : 100kΩ EPCOS (1kΩ pullup)
461 * 52 : 200kΩ ATC Semitec 204GT-2 (1kΩ pullup)
462 * 55 : 100kΩ ATC Semitec 104GT-2 – Used in ParCan & J-Head (1kΩ pullup)
463 *
464 * Analog Thermistors – 10kΩ pullup – Atypical
465 * -----
466 * 99 : 100kΩ Found on some Wanhao i3 machines with a 10kΩ pull-up
resistor
467 *
468 * Analog RTDs (Pt100/Pt1000)
469 * -----
470 * 110 : Pt100 with 1kΩ pullup (atypical)
471 * 147 : Pt100 with 4.7kΩ pullup
472 * 1010 : Pt1000 with 1kΩ pullup (atypical)
473 * 1047 : Pt1000 with 4.7kΩ pullup (E3D)
474 * 20 : Pt100 with circuit in the Ultimainboard V2.x with mainboard ADC
reference voltage = INA826 amplifier-board supply voltage.
475 * NOTE: (1) Must use an ADC input with no pullup. (2) Some
INA826 amplifiers are unreliable at 3.3V so consider using sensor 147, 110,
or 21.
476 * 21 : Pt100 with circuit in the Ultimainboard V2.x with 3.3v ADC
reference voltage (STM32, LPC176x....) and 5V INA826 amplifier board supply.
477 * NOTE: ADC pins are not 5V tolerant. Not recommended
```

```

+/-           NOTE: ADC pins are not yet calibrated, NOT RECOMMENDED
because it's possible to damage the CPU by going over 500°C.
478 *   201 : Pt100 with circuit in Overlord, similar to Ultimainboard V2.x
479 *
480 * Custom/Dummy/Other Thermal Sensors
481 * -----
482 *   0 : not used
483 * 1000 : Custom - Specify parameters in Configuration_adv.h
484 *
485 * !!! Use these for Testing or Development purposes. NEVER for production
machine. !!!
486 * 998 : Dummy Table that ALWAYS reads 25°C or the temperature defined
below.
487 * 999 : Dummy Table that ALWAYS reads 100°C or the temperature defined
below.
488 *
489 */
490 #define TEMP_SENSOR_0 1
491 #define TEMP_SENSOR_1 0
492 #define TEMP_SENSOR_2 0
493 #define TEMP_SENSOR_3 0
494 #define TEMP_SENSOR_4 0
495 #define TEMP_SENSOR_5 0
496 #define TEMP_SENSOR_6 0
497 #define TEMP_SENSOR_7 0
498 #define TEMP_SENSOR_BED 1
499 #define TEMP_SENSOR_PROBE 0
500 #define TEMP_SENSOR_CHAMBER 4
501 #define TEMP_SENSOR_COOLER 0
502 #define TEMP_SENSOR_BOARD 0
503 #define TEMP_SENSOR_REDUNDANT 0
504
505 // Dummy thermistor constant temperature readings, for use with 998 and 999
506 #define DUMMY_THERMISTOR_998_VALUE 25
507 #define DUMMY_THERMISTOR_999_VALUE 100
508
509 // Resistor values when using MAX31865 sensors (-5) on TEMP_SENSOR_0 / 1
510 // #define MAX31865_SENSOR_OHMS_0      100 // (Ω) Typically 100 or 1000
(PT100 or PT1000)
511 // #define MAX31865_CALIBRATION_OHMS_0 430 // (Ω) Typically 430 for
Adafruit PT100; 4300 for Adafruit PT1000
512 // #define MAX31865_SENSOR_OHMS_1      100
513 // #define MAX31865_CALIBRATION_OHMS_1 430
514
515 #define TEMP_RESIDENCY_TIME          10 // (seconds) Time to wait for hotend
to "settle" in M109
516 #define TEMP_WINDOW                  1 // (°C) Temperature proximity for
the "temperature reached" timer
517 #define TEMP_HYSTERESIS              3 // (°C) Temperature proximity
considered "close enough" to the target
518
519 #define TEMP_BED_RESIDENCY_TIME     10 // (seconds) Time to wait for bed to
"settle" in M190
520 #define TEMP_BED_WINDOW             1 // (°C) Temperature proximity for
the "temperature reached" timer
521 #define TEMP_BED_HYSTERESIS         3 // (°C) Temperature proximity
considered "close enough" to the target
522
523 #define TEMP_CHAMBER_RESIDENCY_TIME 10 // (seconds) Time to wait for

```

```

chamber to "settle" in M191
524 #define TEMP_CHAMBER_WINDOW 1 // (°C) Temperature proximity for
the "temperature reached" timer
525 #define TEMP_CHAMBER_HYSTERESIS 3 // (°C) Temperature proximity
considered "close enough" to the target
526
527 /**
528 * Redundant Temperature Sensor (TEMP_SENSOR_REDUNDANT)
529 *
530 * Use a temp sensor as a redundant sensor for another reading. Select an
unused temperature sensor, and another
531 * sensor you'd like it to be redundant for. If the two thermistors differ
by TEMP_SENSOR_REDUNDANT_MAX_DIFF (°C),
532 * the print will be aborted. Whichever sensor is selected will have its
normal functions disabled; i.e. selecting
533 * the Bed sensor (-1) will disable bed heating/monitoring.
534 *
535 * For selecting source/target use: COOLER, PROBE, BOARD, CHAMBER, BED, E0,
E1, E2, E3, E4, E5, E6, E7
536 */
537 #if TEMP_SENSOR_REDUNDANT
538     #define TEMP_SENSOR_REDUNDANT_SOURCE E1 // The sensor that will
provide the redundant reading.
539     #define TEMP_SENSOR_REDUNDANT_TARGET E0 // The sensor that we are
providing a redundant reading for.
540     #define TEMP_SENSOR_REDUNDANT_MAX_DIFF 10 // (°C) Temperature difference
that will trigger a print abort.
541 #endif
542
543 // Below this temperature the heater will be switched off
544 // because it probably indicates a broken thermistor wire.
545 #define HEATER_0_MINTEMP 5
546 #define HEATER_1_MINTEMP 5
547 #define HEATER_2_MINTEMP 5
548 #define HEATER_3_MINTEMP 5
549 #define HEATER_4_MINTEMP 5
550 #define HEATER_5_MINTEMP 5
551 #define HEATER_6_MINTEMP 5
552 #define HEATER_7_MINTEMP 5
553 #define BED_MINTEMP 5
554 #define CHAMBER_MINTEMP 5
555
556 // Above this temperature the heater will be switched off.
557 // This can protect components from overheating, but NOT from shorts and
failures.
558 // (Use MINTEMP for thermistor short/failure protection.)
559 #define HEATER_0_MAXTEMP 275
560 #define HEATER_1_MAXTEMP 275
561 #define HEATER_2_MAXTEMP 275
562 #define HEATER_3_MAXTEMP 275
563 #define HEATER_4_MAXTEMP 275
564 #define HEATER_5_MAXTEMP 275
565 #define HEATER_6_MAXTEMP 275
566 #define HEATER_7_MAXTEMP 275
567 #define BED_MAXTEMP 170
568 #define CHAMBER_MAXTEMP 170
569
570 /**
571 .. Thermal overshoot

```

```

571 * HOTEND_OVERSHOOT
572 * During heatup (and printing) the temperature can often "overshoot" the
573 target by many degrees
574 * (especially before PID tuning). Setting the target temperature too close
575 to MAXTEMP guarantees
576 * a MAXTEMP shutdown! Use these values to forbid temperatures being set too
577 close to MAXTEMP.
578 */
579 #define HOTEND_OVERSHOOT 15 // (°C) Forbid temperatures over MAXTEMP -
580 OVERSHOOT
581 #define BED_OVERSHOOT 10 // (°C) Forbid temperatures over MAXTEMP -
582 OVERSHOOT
583 #define COOLER_OVERSHOOT 2 // (°C) Forbid temperatures closer than
584 OVERSHOOT
585
586 //=====
587 =
588 //===== PID Settings
589 =====
590 =
591 // PID Tuning Guide here: https://reprap.org/wiki/PID\_Tuning
592
593 // Comment the following line to disable PID and enable bang-bang.
594 #define PIDTEMP
595 #define BANG_MAX 255 // Limits current to nozzle while in bang-bang
596 mode; 255=full current
597 #define PID_MAX_BANG_MAX // Limits current to nozzle while PID is active
598 (see PID_FUNCTIONAL_RANGE below); 255=full current
599 #define PID_K1 0.95 // Smoothing factor within any PID loop
600
601 #if ENABLED(PIDTEMP)
602     // #define PID_EDIT_MENU // Add PID editing to the "Advanced
603     // Settings" menu. (~700 bytes of PROGMEM)
604     // #define PID_AUTOTUNE_MENU // Add PID auto-tuning to the "Advanced
605     // Settings" menu. (~250 bytes of PROGMEM)
606     // #define PID_PARAMS_PER_HOTEND // Uses separate PID parameters for each
607     // extruder (useful for mismatched extruders)
608                         // Set/get with gcode: M301 E[extruder
609                         // number, 0-2]
610
611 #if ENABLED(PID_PARAMS_PER_HOTEND)
612     // Specify up to one value per hotend here, according to your setup.
613     // If there are fewer values, the last one applies to the remaining
614     // hotends.
615     #define DEFAULT_Kp_LIST { 22.20, 22.20 }
616     #define DEFAULT_Ki_LIST { 1.08, 1.08 }
617     #define DEFAULT_Kd_LIST { 114.00, 114.00 }
618 #else
619     #define DEFAULT_Kp 30.73
620     #define DEFAULT_Ki 2.61
621     #define DEFAULT_Kd 90.34
622 #endif
623 #endif // PIDTEMP
624
625 //=====
626 =
627 //===== PID > Bed Temperature Control
628 =====

```

```

612 //=====
613 =
614 /**
615 * PID Bed Heating
616 *
617 * If this option is enabled set PID constants below.
618 * If this option is disabled, bang-bang will be used and
619 BED_LIMIT_SWITCHING will enable hysteresis.
620 *
621 * The PID frequency will be the same as the extruder PWM.
622 * If PID_dT is the default, and correct for the hardware/configuration,
623 that means 7.689Hz,
624 * which is fine for driving a square wave into a resistive load and does
625 not significantly
626 * impact FET heating. This also works fine on a Fotek SSR-10DA Solid State
627 Relay into a 250W
628 * heater. If your configuration is significantly different than this and
629 you don't understand
630 * the issues involved, don't use bed PID until someone else verifies that
631 your hardware works.
632 */
633 //#define PIDTEMPBED
634
635 //##define BED_LIMIT_SWITCHING
636
637 /**
638 * Max Bed Power
639 * Applies to all forms of bed control (PID, bang-bang, and bang-bang with
640 hysteresis).
641 * When set to any value below 255, enables a form of PWM to the bed that
642 acts like a divider
643 * so don't use it unless you are OK with PWM on your bed. (See the comment
644 on enabling PIDTEMPBED)
645 */
646 #define MAX_BED_POWER 255 // limits duty cycle to bed; 255=full current
647
648 #if ENABLED(PIDTEMPBED)
649   //#define MIN_BED_POWER 0
650   //#define PID_BED_DEBUG // Sends debug data to the serial port.
651
652   // 120V 250W silicone heater into 4mm borosilicate (MendelMax 1.5+)
653   // from FOPDT model - kp=.39 Tp=405 Tdead=66, Tc set to 79.2, aggressive
654   factor of .15 (vs .1, 1, 10)
655   #define DEFAULT_bedKp 10.00
656   #define DEFAULT_bedKi .023
657   #define DEFAULT_bedKd 305.4
658
659   // FIND YOUR OWN: "M303 E-1 C8 S90" to run autotune on the bed at 90
660   degreesC for 8 cycles.
661 #endif // PIDTEMPBED
662
663 //=====
664 =
665 //===== PID > Chamber Temperature Control
666 =====
667 =

```

```

656 /**
657 * PID Chamber Heating
658 *
659 * If this option is enabled set PID constants below.
660 * If this option is disabled, bang-bang will be used and
661 CHAMBER_LIMIT_SWITCHING will enable
662 * hysteresis.SOFT_PWM_SCALE
663 *
664 * The PID frequency will be the same as the extruder PWM.
665 * If PID_dT is the default, and correct for the hardware/configuration,
666 that means 7.689Hz,
667 * which is fine for driving a square wave into a resistive load and does
668 not significantly
669 * impact FET heating. This also works fine on a Fotek SSR-10DA Solid State
670 Relay into a 200W
671 * heater. If your configuration is significantly different than this and
672 you don't understand
673 * the issues involved, don't use chamber PID until someone else verifies
674 that your hardware works.
675 */
676 #define PIDTEMPCHAMBER
677 // #define CHAMBER_LIMIT_SWITCHING
678
679 /**
680 * Max Chamber Power
681 * Applies to all forms of chamber control (PID, bang-bang, and bang-bang
682 with hysteresis).
683 * When set to any value below 255, enables a form of PWM to the chamber
684 heater that acts like a divider
685 * so don't use it unless you are OK with PWM on your heater. (See the
686 comment on enabling PIDTEMPCHAMBER)
687 */
688 #define MAX_CHAMBER_POWER 255 // limits duty cycle to chamber heater;
689 // 255=full current
690
691 #if ENABLED(PIDTEMPCHAMBER)
692     #define MIN_CHAMBER_POWER 0
693     // #define PID_CHAMBER_DEBUG // Sends debug data to the serial port.
694
695     // Lasko "MyHeat Personal Heater" (200w) modified with a Fotek SSR-10DA to
696     // control only the heating element
697     // and placed inside the small Creality printer enclosure tent.
698     //
699     #define DEFAULT_chamberKp 37.04
700     #define DEFAULT_chamberKi 1.40
701     #define DEFAULT_chamberKd 655.17
702     // M309 P37.04 I1.04 D655.17
703
704     // FIND YOUR OWN: "M303 E-2 C8 S50" to run autotune on the chamber at 50
705 degreesC for 8 cycles.
706 #endif // PIDTEMPCHAMBER
707
708 #if ANY(PIDTEMP, PIDTEMPBED, PIDTEMPCHAMBER)
709     // #define PID_DEBUG // Sends debug data to the serial port.
710     Use 'M303 D' to toggle activation.
711     // #define PID_OPENLOOP // Puts PID in open loop. M104/M140 sets
712     the output power from 0 to PID_MAX
713     // #define SLOW_PWM_HEATERS // PWM with very low frequency (roughly

```

```
0.125Hz=8s) and minimum state time of approximately 1s useful for heaters  
driven by a relay  
700 #define PID_FUNCTIONAL_RANGE 10 // If the temperature difference between  
the target temperature and the actual temperature  
701 // is more than PID_FUNCTIONAL_RANGE then  
the PID will be shut off and the heater will be set to min/max.  
702 #endif  
703  
704 // @section extruder  
705  
706 /**  
 * Prevent extrusion if the temperature is below EXTRUDE_MINTEMP.  
 * Add M302 to set the minimum extrusion temperature and/or turn  
 * cold extrusion prevention on and off.  
 *  
 * *** IT IS HIGHLY RECOMMENDED TO LEAVE THIS OPTION ENABLED! ***  
 */  
713 #define PREVENT_COLD_EXTRUSION  
714 #define EXTRUDE_MINTEMP 140  
715  
716 /**  
 * Prevent a single extrusion longer than EXTRUDE_MAXLENGTH.  
 * Note: For Bowden Extruders make this large enough to allow load/unload.  
 */  
720 #define PREVENT_LENGTHY_EXTRUDE  
721 #define EXTRUDE_MAXLENGTH 200  
722  
723 //=====  
=  
724 //===== Thermal Runaway Protection  
=====  
725 //=====  
=  
726  
727 /**  
 * Thermal Protection provides additional protection to your printer from  
damage  
729 * and fire. Marlin always includes safe min and max temperature ranges  
which  
730 * protect against a broken or disconnected thermistor wire.  
*  
732 * The issue: If a thermistor falls out, it will report the much lower  
733 * temperature of the air in the room, and the the firmware will keep  
734 * the heater on.  
*  
736 * If you get "Thermal Runaway" or "Heating failed" errors the  
737 * details can be tuned in Configuration_adv.h  
*/  
739  
740 #define THERMAL_PROTECTION_HOTENDS // Enable thermal protection for all  
extruders  
741 #define THERMAL_PROTECTION_BED // Enable thermal protection for the  
heated bed  
742 #define THERMAL_PROTECTION_CHAMBER // Enable thermal protection for the  
heated chamber  
743 #define THERMAL_PROTECTION_COOLER // Enable thermal protection for the  
laser cooling  
744 ..
```

```
//=====
//=====
746 //===== Mechanical Settings
//=====
747 //=====
//=====
748
749 // @section machine
750
751 // Enable one of the options below for CoreXY, CoreXZ, or CoreYZ kinematics,
752 // either in the usual order or reversed
753 // #define COREXY
754 // #define COREXZ
755 // #define COREYZ
756 // #define COREYX
757 // #define COREZX
758 // #define COREZY
759 // #define MARKFORGED_XY // MarkForged. See
// https://reprap.org/forum/read.php?152,504042
760
761 // Enable for a belt style printer with endless "Z" motion
762 // #define BELTPRINTER
763
764 // Enable for Polargraph Kinematics
765 // #define POLARGRAPH
766 #if ENABLED(POLARGRAPH)
767   #define POLARGRAPH_MAX_BELT_LEN 1035.0
768   #define POLAR_SEGMENTS_PER_SECOND 5
769 #endif
770
771 //=====
//=====
772 //===== Endstop Settings
//=====
773 //=====
//=====
774
775 // @section homing
776
777 // Specify here all the endstop connectors that are connected to any endstop
// or probe.
778 // Almost all printers will be using one per axis. Probes will use one or
// more of the
779 // extra connectors. Leave undefined any used for non-endstop and non-probe
// purposes.
780 // #define USE_XMIN_PLUG
781 // #define USE_YMIN_PLUG
782 #define USE_ZMIN_PLUG
783 // #define USE_IMIN_PLUG
784 // #define USE_JMIN_PLUG
785 // #define USE_KMIN_PLUG
786 #define USE_XMAX_PLUG
787 #define USE_YMAX_PLUG
788 // #define USE_ZMAX_PLUG
789 // #define USE_IMAX_PLUG
790 // #define USE_JMAX_PLUG
791 // #define USE_KMAX_PLUG
792
793 // Enable pullup for all endstops to prevent a floating state
```

```
794 #define ENDSTOPPULLUPS
795 #if DISABLED(ENDSTOPPULLUPS)
796     // Disable ENDSTOPPULLUPS to set pullups individually
797     //#define ENDSTOPPULLUP_XMAX
798     //#define ENDSTOPPULLUP_YMAX
799     //#define ENDSTOPPULLUP_ZMAX
800     //#define ENDSTOPPULLUP_IMAX
801     //#define ENDSTOPPULLUP_JMAX
802     //#define ENDSTOPPULLUP_KMAX
803     //#define ENDSTOPPULLUP_XMIN
804     //#define ENDSTOPPULLUP_YMIN
805     //#define ENDSTOPPULLUP_ZMIN
806     //#define ENDSTOPPULLUP_IMIN
807     //#define ENDSTOPPULLUP_JMIN
808     //#define ENDSTOPPULLUP_KMIN
809     //#define ENDSTOPPULLUP_ZMIN_PROBE
810 #endif
811
812 // Enable pulldown for all endstops to prevent a floating state
813 //#define ENDSTOPPULLDOWNS
814 #if DISABLED(ENDSTOPPULLDOWNS)
815     // Disable ENDSTOPPULLDOWNS to set pulldowns individually
816     //#define ENDSTOPPULLDOWN_XMAX
817     //#define ENDSTOPPULLDOWN_YMAX
818     //#define ENDSTOPPULLDOWN_ZMAX
819     //#define ENDSTOPPULLDOWN_IMAX
820     //#define ENDSTOPPULLDOWN_JMAX
821     //#define ENDSTOPPULLDOWN_KMAX
822     //#define ENDSTOPPULLDOWN_XMIN
823     //#define ENDSTOPPULLDOWN_YMIN
824     //#define ENDSTOPPULLDOWN_ZMIN
825     //#define ENDSTOPPULLDOWN_IMIN
826     //#define ENDSTOPPULLDOWN_JMIN
827     //#define ENDSTOPPULLDOWN_KMIN
828     //#define ENDSTOPPULLDOWN_ZMIN_PROBE
829 #endif
830
831 // Mechanical endstop with COM to ground and NC to Signal uses "false" here
// (most common setup).
832 #define X_MIN_ENDSTOP_INVERTING false // Set to true to invert the logic of
// the endstop.
833 #define Y_MIN_ENDSTOP_INVERTING false // Set to true to invert the logic of
// the endstop.
834 #define Z_MIN_ENDSTOP_INVERTING false // Set to true to invert the logic of
// the endstop.
835 #define I_MIN_ENDSTOP_INVERTING false // Set to true to invert the logic of
// the endstop.
836 #define J_MIN_ENDSTOP_INVERTING false // Set to true to invert the logic of
// the endstop.
837 #define K_MIN_ENDSTOP_INVERTING false // Set to true to invert the logic of
// the endstop.
838 #define X_MAX_ENDSTOP_INVERTING false // Set to true to invert the logic of
// the endstop.
839 #define Y_MAX_ENDSTOP_INVERTING false // Set to true to invert the logic of
// the endstop.
840 #define Z_MAX_ENDSTOP_INVERTING false // Set to true to invert the logic of
// the endstop.
841 #define I_MAX_ENDSTOP_INVERTING false // Set to true to invert the logic of
```

```
tne enstop.
842 #define J_MAX_ENDSTOP_INVERTING false // Set to true to invert the logic of
the endstop.
843 #define K_MAX_ENDSTOP_INVERTING false // Set to true to invert the logic of
the endstop.
844 #define Z_MIN_PROBE_ENDSTOP_INVERTING false // Set to true to invert the
logic of the probe.
845
846 /**
847 * Stepper Drivers
848 *
849 * These settings allow Marlin to tune stepper driver timing and enable
advanced options for
850 * stepper drivers that support them. You may also override timing options
in Configuration_adv.h.
851 *
852 * A4988 is assumed for unspecified drivers.
853 *
854 * Use TMC2208/TMC2208_STANDALONE for TMC2225 drivers and
TMC2209/TMC2209_STANDALONE for TMC2226 drivers.
855 *
856 * Options: A4988, A5984, DRV8825, LV8729, L6470, L6474, POWERSTEP01,
857 * TB6560, TB6600, TMC2100,
858 * TMC2130, TMC2130_STANDALONE, TMC2160, TMC2160_STANDALONE,
859 * TMC2208, TMC2208_STANDALONE, TMC2209, TMC2209_STANDALONE,
860 * TMC26X, TMC26X_STANDALONE, TMC2660, TMC2660_STANDALONE,
861 * TMC5130, TMC5130_STANDALONE, TMC5160, TMC5160_STANDALONE
862 * :['A4988', 'A5984', 'DRV8825', 'LV8729', 'L6470', 'L6474', 'POWERSTEP01',
'TB6560', 'TB6600', 'TMC2100', 'TMC2130', 'TMC2130_STANDALONE', 'TMC2160',
'TMC2160_STANDALONE', 'TMC2208', 'TMC2208_STANDALONE', 'TMC2209',
'TMC2209_STANDALONE', 'TMC26X', 'TMC26X_STANDALONE', 'TMC2660',
'TMC2660_STANDALONE', 'TMC5130', 'TMC5130_STANDALONE', 'TMC5160',
'TMC5160_STANDALONE']
863 */
864 #define X_DRIVER_TYPE DRV8825
865 #define Y_DRIVER_TYPE DRV8825
866 #define Z_DRIVER_TYPE DRV8825
867 //#define X2_DRIVER_TYPE A4988
868 //#define Y2_DRIVER_TYPE A4988
869 //#define Z2_DRIVER_TYPE A4988
870 //#define Z3_DRIVER_TYPE A4988
871 //#define Z4_DRIVER_TYPE A4988
872 //#define I_DRIVER_TYPE A4988
873 //#define J_DRIVER_TYPE A4988
874 //#define K_DRIVER_TYPE A4988
875 #define E0_DRIVER_TYPE DRV8825
876 //#define E1_DRIVER_TYPE A4988
877 //#define E2_DRIVER_TYPE A4988
878 //#define E3_DRIVER_TYPE A4988
879 //#define E4_DRIVER_TYPE A4988
880 //#define E5_DRIVER_TYPE A4988
881 //#define E6_DRIVER_TYPE A4988
882 //#define E7_DRIVER_TYPE A4988
883
884 // Enable this feature if all enabled endstop pins are interrupt-capable.
885 // This will remove the need to poll the interrupt pins, saving many CPU
cycles.
886 //#define ENDSTOP_INTERRUPTS_FEATURE
887
```

```

888 /**
889 * Endstop Noise Threshold
890 *
891 * Enable if your probe or endstops falsely trigger due to noise.
892 *
893 * - Higher values may affect repeatability or accuracy of some bed probes.
894 * - To fix noise install a 100nF ceramic capacitor in parallel with the
895 * switch.
896 * - This feature is not required for common micro-switches mounted on PCBs
897 * based on the Makerbot design, which already have the 100nF capacitor.
898 *
899 * :[2,3,4,5,6,7]
900 */
901 //#define ENDSTOP_NOISE_THRESHOLD 2
902
903 // Check for stuck or disconnected endstops during homing moves.
904 //#define DETECT_BROKEN_ENDSTOP
905
906 //=====
907 //=====
908 // @section motion
909
910 /**
911 * Default Settings
912 *
913 * These settings can be reset by M502
914 *
915 * Note that if EEPROM is enabled, saved values will override these.
916 */
917
918 /**
919 * With this option each E stepper can have its own factors for the
920 * following movement settings. If fewer factors are given than the
921 * total number of extruders, the last value applies to the rest.
922 */
923 //#define DISTINCT_E_FACTORS
924
925 /**
926 * Default Axis Steps Per Unit (steps/mm)
927 * Override with M92
928 *
929 * X, Y, Z [, I [, J [, K]]], E0 [, E1[, E2...]]
930 */
931 #define DEFAULT_AXIS_STEPS_PER_UNIT { 160, 160, 1600, 829.23 }
932
933 /**
934 * Default Max Feed Rate (mm/s)
935 * Override with M203
936 *
937 * X, Y, Z [, I [, J [, K]]], E0 [, E1[, E2...]]
938 */
939 #define DEFAULT_MAX_FEEDRATE { 500, 500, 50, 25 }
940
941 // #define LIMITED_MAX_FR_EDITING // Limit edit via M203 or LCD to

```

```

940 #define DEFAULT_MAX_FEEDRATE * 2
941 #if ENABLED(LIMITED_MAX_FR_EDITING)
942     #define MAX_FEEDRATE_EDIT_VALUES      { 600, 600, 10, 50 } // ...or, set
943     your own edit limits
944 #endif
945
946 /**
947  * Default Max Acceleration (change/s) change = mm/s
948  * (Maximum start speed for accelerated moves)
949  * Override with M201
950  *
951  *          X, Y, Z [, I [, J [, K]]], E0 [, E1[, E2...]]
952  */
953 #define DEFAULT_MAX_ACCELERATION      { 500, 500, 100, 5000 }
954
955 // #define LIMITED_MAX_ACCEL_EDITING // Limit edit via M201 or LCD to
956 #define DEFAULT_MAX_ACCELERATION * 2
957 #if ENABLED(LIMITED_MAX_ACCEL_EDITING)
958     #define MAX_ACCEL_EDIT_VALUES        { 6000, 6000, 200, 20000 } // ...or,
959     set your own edit limits
960 #endif
961
962 /**
963  * Default Acceleration (change/s) change = mm/s
964  * Override with M204
965  *
966  *      M204 P    Acceleration
967  *      M204 R    Retract Acceleration
968  *      M204 T    Travel Acceleration
969  */
970 #define DEFAULT_ACCELERATION          500 // X, Y, Z and E acceleration
971 for printing moves
972 #define DEFAULT_RETRACT_ACCELERATION 500 // E acceleration for retracts
973 #define DEFAULT_TRAVEL_ACCELERATION  500 // X, Y, Z acceleration for
974 travel (non printing) moves
975
976 /**
977  * Default Jerk limits (mm/s)
978  * Override with M205 X Y Z E
979  *
980  * "Jerk" specifies the minimum speed change that requires acceleration.
981  * When changing speed and direction, if the difference is less than the
982  * value set here, it may happen instantaneously.
983  */
984 // #define CLASSIC_JERK
985 #if ENABLED(CLASSIC_JERK)
986     #define DEFAULT_XJERK 10.0
987     #define DEFAULT_YJERK 10.0
988     #define DEFAULT_ZJERK  0.3
989     // #define DEFAULT_IJERK 0.3
990     // #define DEFAULT_JJERK 0.3
991     // #define DEFAULT_KJERK 0.3
992
993     // #define TRAVEL_EXTRA_XYJERK 0.0 // Additional jerk allowance for all
994 travel moves
995
996 // #define LIMITED_JERK_EDITING // Limit edit via M205 or LCD to
997 #define DEFAULT_aJERK * 2
998 #if FNARI FD(I TMTTFD 1FRK FDTTTNG)

```

```

990     #define MAX_JERK_EDIT_VALUES { 20, 20, 0.6, 10 } // ...or, set your own
991     edit limits
992 #endif
993
994 #define DEFAULT_EJERK      5.0 // May be used by Linear Advance
995
996 /**
997  * Junction Deviation Factor
998  *
999  * See:
1000 *   https://reprap.org/forum/read.php?1,739819
1001 *   https://blog.kyneticcnc.com/2018/10/computing-junction-deviation-for-marlin.html
1002 */
1003 #if DISABLED(CLASSIC_JERK)
1004     #define JUNCTION_DEVIATION_MM 0.013 // (mm) Distance from real junction
edge
1005     #define JD_HANDLE_SMALL_SEGMENTS      // Use curvature estimation instead of
just the junction angle                      // for small segments (< 1mm) with
large junction angles (> 135°).
1006 #endif
1007
1008 /**
1009  * S-Curve Acceleration
1010  *
1011  * This option eliminates vibration during printing by fitting a Bézier
1012  * curve to move acceleration, producing much smoother direction changes.
1013  *
1014  * See https://github.com/synthetos/TinyG/wiki/Jerk-Controlled-Motion-Explained
1015 */
1016 // #define S_CURVE_ACCELERATION
1017
1018 //=====
1019 =
1020 //===== Z Probe Options
1021 =====
1022 =
1023 // @section probes
1024 //
1025 // See https://marlinfw.org/docs/configuration/probes.html
1026 //
1027
1028 /**
1029  * Enable this option for a probe connected to the Z-MIN pin.
1030  * The probe replaces the Z-MIN endstop and is used for Z homing.
1031  * (Automatically enables USE_PROBE_FOR_Z_HOMING.)
1032 */
1033 // #define Z_MIN_PROBEUSES_Z_MIN_ENDSTOP_PIN
1034
1035 // Force the use of the probe for Z-axis homing
1036 // #define USE_PROBE_FOR_Z_HOMING
1037
1038 /**

```

```
1039 * Z_MIN_PROBE_PIN
1040 *
1041 * Define this pin if the probe is not connected to Z_MIN_PIN.
1042 * If not defined the default pin for the selected MOTHERBOARD
1043 * will be used. Most of the time the default is what you want.
1044 *
1045 * - The simplest option is to use a free endstop connector.
1046 * - Use 5V for powered (usually inductive) sensors.
1047 *
1048 * - RAMPS 1.3/1.4 boards may use the 5V, GND, and Aux4->D32 pin:
1049 *   - For simple switches connect...
1050 *     - normally-closed switches to GND and D32.
1051 *     - normally-open switches to 5V and D32.
1052 */
1053 //#define Z_MIN_PROBE_PIN 32 // Pin 32 is the RAMPS default
1054
1055 /**
1056 * Probe Type
1057 *
1058 * Allen Key Probes, Servo Probes, Z-Sled Probes, FIX_MOUNTED_PROBE, etc.
1059 * Activate one of these to use Auto Bed Leveling below.
1060 */
1061
1062 /**
1063 * The "Manual Probe" provides a means to do "Auto" Bed Leveling without a
1064 * probe.
1065 * Use G29 repeatedly, adjusting the Z height at each point with movement
1066 * commands
1067 * or (with LCD_BED_LEVELING) the LCD controller.
1068 */
1069 //#define PROBE_MANUALLY
1070
1071 /**
1072 * A Fix-Mounted Probe either doesn't deploy or needs manual deployment.
1073 * (e.g., an inductive probe or a nozzle-based probe-switch.)
1074 */
1075 //#define FIX_MOUNTED_PROBE
1076
1077 /**
1078 * Use the nozzle as the probe, as with a conductive
1079 * nozzle system or a piezo-electric smart effector.
1080 */
1081 //#define NOZZLE_AS_PROBE
1082
1083 /**
1084 * Z Servo Probe, such as an endstop switch on a rotating arm.
1085 */
1086 //#define Z_PROBE_SERVO_NR 0      // Defaults to SERVO 0 connector.
1087 //#define Z_SERVO_ANGLES { 70, 0 } // Z Servo Deploy and Stow angles
1088
1089 /**
1090 * The BLTouch probe uses a Hall effect sensor and emulates a servo.
1091 */
1092 //#define BLTOUCH
1093
1094 /**
1095 * Touch-MI Probe by hotends.fr
1096 *
1097 * This probe is deployed and activated by moving the X-axis to a magnet at
```

```

1000 * This probe is deployed and activated by moving the Z axis to a magnet at
1001 the edge of the bed.
1002 * By default, the magnet is assumed to be on the left and activated by a
1003 home. If the magnet is
1004 * on the right, enable and set TOUCH_MI_DEPLOY_XPOS to the deploy position.
1005 *
1006 * Also requires: BABYSTEPPING, BABYSTEP_ZPROBE_OFFSET, Z_SAFE_HOMING,
1007 * and a minimum Z_HOME_HEIGHT of 10.
1008 */
1009 // #define TOUCH_MI_PROBE
1010 #if ENABLED(TOUCH_MI_PROBE)
1011   #define TOUCH_MI_RETRACT_Z 0.5                                // Height at which the
1012   probe retracts
1013   // #define TOUCH_MI_DEPLOY_XPOS (X_MAX_BED + 2) // For a magnet on the
1014   right side of the bed
1015   // #define TOUCH_MI_MANUAL_DEPLOY                         // For manual deploy (LCD
1016   menu)
1017 #endif
1018
1019 // A probe that is deployed and stowed with a solenoid pin (SOL1_PIN)
1020 // #define SOLENOID_PROBE
1021
1022 // A sled-mounted probe like those designed by Charles Bell.
1023 // #define Z_PROBE_SLED
1024 // #define SLED_DOCKING_OFFSET 5 // The extra distance the X axis must
1025 travel to pickup the sled. 0 should be fine but you can push it further if
1026 you'd like.
1027
1028 // A probe deployed by moving the x-axis, such as the Wilson II's rack-and-
1029 pinion probe designed by Marty Rice.
1030 // #define RACK_AND_PINION_PROBE
1031 #if ENABLED(RACK_AND_PINION_PROBE)
1032   #define Z_PROBE_DEPLOY_X X_MIN_POS
1033   #define Z_PROBE_RETRACT_X X_MAX_POS
1034 #endif
1035
1036 // Duet Smart Effector (for delta printers) - https://bit.ly/2uL5U7J
1037 // When the pin is defined you can use M672 to set/reset the probe
1038 sensitivity.
1039 // #define DUET_SMART_EFFECTOR
1040 #if ENABLED(DUET_SMART_EFFECTOR)
1041   #define SMART_EFFECTOR_MOD_PIN -1 // Connect a GPIO pin to the Smart
1042 Effect or MOD pin
1043 #endif
1044
1045 /**
1046 * Use StallGuard2 to probe the bed with the nozzle.
1047 * Requires stallGuard-capable Trinamic stepper drivers.
1048 * CAUTION: This can damage machines with Z lead screws.
1049 *          Take extreme care when setting up this feature.
1050 */
1051 // #define SENSORLESS_PROBING
1052
1053 /**
1054 * Nozzle-to-Probe offsets { X, Y, Z }
1055 */

```

```

1144 *
1145 * X and Y offset
1146 * Use a caliper or ruler to measure the distance from the tip of
1147 * the Nozzle to the center-point of the Probe in the X and Y axes.
1148 *
1149 * Z offset
1150 * - For the Z offset use your best known value and adjust at runtime.
1151 * - Common probes trigger below the nozzle and have negative values for Z
1152 offset.
1153 * - Probes triggering above the nozzle height are uncommon but do exist.
When using
1153 * probes such as this, carefully set Z_CLEARANCE_DEPLOY_PROBE and
Z_CLEARANCE_BETWEEN_PROBES
1154 * to avoid collisions during probing.
1155 *
1156 * Tune and Adjust
1157 * - Probe Offsets can be tuned at runtime with 'M851', LCD menus,
babystepping, etc.
1158 * - PROBE_OFFSET_WIZARD (configuration_adv.h) can be used for setting the
Z offset.
1159 *
1160 * Assuming the typical work area orientation:
1161 * - Probe to RIGHT of the Nozzle has a Positive X offset
1162 * - Probe to LEFT of the Nozzle has a Negative X offset
1163 * - Probe in BACK of the Nozzle has a Positive Y offset
1164 * - Probe in FRONT of the Nozzle has a Negative Y offset
1165 *
1166 * Some examples:
1167 * #define NOZZLE_TO_PROBE_OFFSET { 10, 10, -1 } // Example "1"
1168 * #define NOZZLE_TO_PROBE_OFFSET {-10, 5, -1 } // Example "2"
1169 * #define NOZZLE_TO_PROBE_OFFSET { 5, -5, -1 } // Example "3"
1170 * #define NOZZLE_TO_PROBE_OFFSET {-15,-10, -1 } // Example "4"
1171 *
1172 *      +--- BACK ---+
1173 *      | [+]   |
1174 *      L |       1 | R <-- Example "1" (right+, back+)
1175 *      E | 2     | I <-- Example "2" ( left-, back+)
1176 *      F | [-] N | [+]| G <-- Nozzle
1177 *      T |       3 | H <-- Example "3" (right+, front-)
1178 *      | 4     | T <-- Example "4" ( left-, front-)
1179 *      | [-]   |
1180 *      0-- FRONT --+
1181 */
1182 #define NOZZLE_TO_PROBE_OFFSET { 10, 10, 0 }
1183
1184 // Most probes should stay away from the edges of the bed, but
1185 // with NOZZLE_AS_PROBE this can be negative for a wider probing area.
1186 #define PROBING_MARGIN 10
1187
1188 // X and Y axis travel speed (mm/min) between probes
1189 #define XY_PROBE_FEEDRATE (133*60)
1190
1191 // Feedrate (mm/min) for the first approach when double-probing
(MULTIPLE_PROBING == 2)
1192 #define Z_PROBE_FEEDRATE_FAST (4*60)
1193
1194 // Feedrate (mm/min) for the "accurate" probe of each point
1195 #define Z_PROBE_FEEDRATE_SLOW (Z_PROBE_FEEDRATE_FAST / 2)
1196

```

```

1197 /**
1198 * Probe Activation Switch
1199 * A switch indicating proper deployment, or an optical
1200 * switch triggered when the carriage is near the bed.
1201 */
1202 // #define PROBE_ACTIVATION_SWITCH
1203 #if ENABLED(PROBE_ACTIVATION_SWITCH)
1204     #define PROBE_ACTIVATION_SWITCH_STATE LOW // State indicating probe is
1205     active
1206     // #define PROBE_ACTIVATION_SWITCH_PIN PC6 // Override default pin
1207 #endif
1208 /**
1209 * Tare Probe (determine zero-point) prior to each probe.
1210 * Useful for a strain gauge or piezo sensor that needs to factor out
1211 * elements such as cables pulling on the carriage.
1212 */
1213 // #define PROBE_TARE
1214 #if ENABLED(PROBE_TARE)
1215     #define PROBE_TARE_TIME 200      // (ms) Time to hold tare pin
1216     #define PROBE_TARE_DELAY 200    // (ms) Delay after tare before
1217     #define PROBE_TARE_STATE HIGH   // State to write pin for tare
1218     // #define PROBE_TARE_PIN PA5    // Override default pin
1219     #if ENABLED(PROBE_ACTIVATION_SWITCH)
1220         // #define PROBE_TARE_ONLY_WHILE_INACTIVE // Fail to tare/probe if
1221         PROBE_ACTIVATION_SWITCH is active
1222     #endif
1223 #endif
1224 /**
1225 * Multiple Probing
1226 *
1227 * You may get improved results by probing 2 or more times.
1228 * With EXTRA_PROBING the more atypical reading(s) will be disregarded.
1229 *
1230 * A total of 2 does fast/slow probes with a weighted average.
1231 * A total of 3 or more adds more slow probes, taking the average.
1232 */
1233 // #define MULTIPLE_PROBING 2
1234 // #define EXTRA_PROBING    1
1235
1236 /**
1237 * Z probes require clearance when deploying, stowing, and moving between
1238 * probe points to avoid hitting the bed and other hardware.
1239 * Servo-mounted probes require extra space for the arm to rotate.
1240 * Inductive probes need space to keep from triggering early.
1241 *
1242 * Use these settings to specify the distance (mm) to raise the probe (or
1243 * lower the bed). The values set here apply over and above any (negative)
1244 * probe Z Offset set with NOZZLE_TO_PROBE_OFFSET, M851, or the LCD.
1245 * Only integer values >= 1 are valid here.
1246 *
1247 * Example: `M851 Z-5` with a CLEARANCE of 4 => 9mm from bed to nozzle.
1248 * But: `M851 Z+1` with a CLEARANCE of 2 => 2mm from bed to nozzle.
1249 */
1250 #define Z_CLEARANCE_DEPLOY_PROBE 10 // Z Clearance for Deploy/Stow
1251 #define Z_CLEARANCE_BETWEEN_PROBES 5 // Z Clearance between probe points
1252 #define Z_CLEARANCE_MULTI_PROBE 5 // Z Clearance between multiple probes

```

```

1253 // #define Z_AFTER_PROBING           5 // Z position after probing is done
1254
1255 #define Z_PROBE_LOW_POINT         -2 // Farthest distance below the
1256   trigger-point to go before stopping
1257
1258 // For M851 give a range for adjusting the Z probe offset
1259 #define Z_PROBE_OFFSET_RANGE_MIN -20
1260 #define Z_PROBE_OFFSET_RANGE_MAX  20
1261
1262 // Enable the M48 repeatability test to test probe accuracy
1263 //#define Z_MIN_PROBE_REPEATABILITY_TEST
1264
1265 // Before deploy/stow pause for user confirmation
1266 //#define PAUSE_BEFORE_DEPLOY_STOW
1267 #if ENABLED(PAUSE_BEFORE_DEPLOY_STOW)
1268   // #define PAUSE_PROBE_DEPLOY_WHEN_TRIGGERED // For Manual Deploy Allenkey
1269   Probe
1270 #endif
1271
1272 /**
1273  * Enable one or more of the following if probing seems unreliable.
1274  * Heaters and/or fans can be disabled during probing to minimize electrical
1275  * noise. A delay can also be added to allow noise and vibration to settle.
1276  * These options are most useful for the BLTouch probe, but may also improve
1277  * readings with inductive probes and piezo sensors.
1278 */
1279 //#define PROBING_HEATERS_OFF          // Turn heaters off when probing
1280 #if ENABLED(PROBING_HEATERS_OFF)
1281   // #define WAIT_FOR_BED_HEATER        // Wait for bed to heat back up between
1282   probes (to improve accuracy)
1283   // #define WAIT_FOR_HOTEND          // Wait for hotend to heat back up
1284   between probes (to improve accuracy & prevent cold extrude)
1285 #endif
1286 //#define PROBING_FANS_OFF            // Turn fans off when probing
1287 //#define PROBING_ESTEPERS_OFF        // Turn all extruder steppers off when
1288   probing
1289 //#define PROBING_STEPPERS_OFF        // Turn all steppers off (unless needed
1290   to hold position) when probing (including extruders)
1291 //#define DELAY_BEFORE_PROBING 200 // (ms) To prevent vibrations from
1292   triggering piezo sensors
1293
1294 // Require minimum nozzle and/or bed temperature for probing
1295 //#define PREHEAT_BEFORE_PROBING
1296 #if ENABLED(PREHEAT_BEFORE_PROBING)
1297   #define PROBING_NOZZLE_TEMP 120 // (°C) Only applies to E0 at this time
1298   #define PROBING_BED_TEMP      50
1299 #endif
1300
1301 // For Inverting Stepper Enable Pins (Active Low) use 0, Non Inverting
1302 // (Active High) use 1
1303 // :{ 0:'Low', 1:'High' }
1304 #define X_ENABLE_ON 0
1305 #define Y_ENABLE_ON 0
1306 #define Z_ENABLE_ON 0
1307 #define E_ENABLE_ON 0 // For all extruders
1308 // #define I_ENABLE_ON 0
1309 // #define J_ENABLE_ON 0
1310 // #define K_ENABLE_ON 0

```

```
1304 // Disable axis steppers immediately when they're not being stepped.  
1305 // WARNING: When motors turn off there is a chance of losing position  
1306 // accuracy!  
1307 #define DISABLE_X false  
1308 #define DISABLE_Y false  
1309 #define DISABLE_Z false  
1310 //##define DISABLE_I false  
1311 //##define DISABLE_J false  
1312 //##define DISABLE_K false  
1313  
1314 // Turn off the display blinking that warns about possible accuracy  
1315 // reduction  
1316 //##define DISABLE_REDUCED_ACCURACY_WARNING  
1317  
1318 // @section extruder  
1319  
1320 #define DISABLE_E false // Disable the extruder when not  
1321 // stepping  
1322 #define DISABLE_INACTIVE_EXTRUDER // Keep only the active extruder enabled  
1323  
1324 // @section machine  
1325  
1326 // Invert the stepper direction. Change (or reverse the motor connector) if  
1327 // an axis goes the wrong way.  
1328 #define INVERT_X_DIR false  
1329 #define INVERT_Y_DIR false  
1330 #define INVERT_Z_DIR false  
1331 //##define INVERT_I_DIR false  
1332 //##define INVERT_J_DIR false  
1333 //##define INVERT_K_DIR false  
1334  
1335 // @section extruder  
1336  
1337 // For direct drive extruder v9 set to true, for geared extruder set to  
1338 // false.  
1339 #define INVERT_E0_DIR true  
1340 #define INVERT_E1_DIR false  
1341 #define INVERT_E2_DIR false  
1342 #define INVERT_E3_DIR false  
1343 #define INVERT_E4_DIR false  
1344 #define INVERT_E5_DIR false  
1345 #define INVERT_E6_DIR false  
1346 #define INVERT_E7_DIR false  
1347  
1348 // @section homing  
1349  
1350 //##define NO_MOTION_BEFORE_HOMING // Inhibit movement until all axes have  
1351 // been homed. Also enable HOME_AFTER_DEACTIVATE for extra safety.  
1352 //##define HOME_AFTER_DEACTIVATE // Require rehoming after steppers are  
1353 // deactivated. Also enable NO_MOTION_BEFORE_HOMING for extra safety.  
1354  
1355 /**  
1356 * Set Z_IDLE_HEIGHT if the Z-Axis moves on its own when steppers are  
1357 // disabled.  
1358 * - Use a low value (i.e., Z_MIN_POS) if the nozzle falls down to the bed.  
1359 * - Use a large value (i.e., Z_MAX_POS) if the bed falls down, away from  
1360 // the nozzle.  
1361 */
```

```

1353 // #define Z_IDLE_HEIGHT Z_HOME_POS
1354
1355 // #define Z_HOMING_HEIGHT 4          // (mm) Minimal Z height before homing
1356 // (G28) for Z clearance above the bed, clamps, ...
1357 //                                         // Be sure to have this much clearance
1358 // over your Z_MAX_POS to prevent grinding.
1359
1360 // #define Z_AFTER_HOMING 10        // (mm) Height to move to after homing Z
1361
1362 // Direction of endstops when homing; 1=MAX, -1=MIN
1363 // :[-1,1]
1364 #define X_HOME_DIR 1
1365 #define Y_HOME_DIR 1
1366 #define Z_HOME_DIR -1
1367 // #define I_HOME_DIR -1
1368 // #define J_HOME_DIR -1
1369 // #define K_HOME_DIR -1
1370
1371 // @section machine
1372
1373 // The size of the printable area
1374 #define X_BED_SIZE 230
1375 #define Y_BED_SIZE 225
1376
1377 // Travel limits (mm) after homing, corresponding to endstop positions.
1378 #define X_MIN_POS 0
1379 #define Y_MIN_POS 0
1380 #define Z_MIN_POS 0
1381 #define X_MAX_POS X_BED_SIZE
1382 #define Y_MAX_POS Y_BED_SIZE
1383 #define Z_MAX_POS 300
1384 // #define I_MIN_POS 0
1385 // #define I_MAX_POS 50
1386 // #define J_MIN_POS 0
1387 // #define J_MAX_POS 50
1388 // #define K_MIN_POS 0
1389 // #define K_MAX_POS 50
1390
1391 /**
1392 * Software Endstops
1393 *
1394 * - Prevent moves outside the set machine bounds.
1395 * - Individual axes can be disabled, if desired.
1396 * - X and Y only apply to Cartesian robots.
1397 * - Use 'M211' to set software endstops on/off or report current state
1398 */
1399
1400 // Min software endstops constrain movement within minimum coordinate bounds
1401 #define MIN_SOFTWARE_ENDSTOPS
1402 #if ENABLED(MIN_SOFTWARE_ENDSTOPS)
1403     #define MIN_SOFTWARE_ENDSTOP_X
1404     #define MIN_SOFTWARE_ENDSTOP_Y
1405     #define MIN_SOFTWARE_ENDSTOP_Z
1406     #define MIN_SOFTWARE_ENDSTOP_I
1407     #define MIN_SOFTWARE_ENDSTOP_J
1408     #define MIN_SOFTWARE_ENDSTOP_K
1409 #endif
1410
1411 // Max software endstops constrain movement within maximum coordinate bounds

```

```
// MAX SOFTWARE ENDSTOPS CONFIGURATION. movement within maximum coordinate bounds
1410 #define MAX_SOFTWARE_ENDSTOPS
1411 #if ENABLED(MAX_SOFTWARE_ENDSTOPS)
1412     #define MAX_SOFTWARE_ENDSTOP_X
1413     #define MAX_SOFTWARE_ENDSTOP_Y
1414     #define MAX_SOFTWARE_ENDSTOP_Z
1415     #define MAX_SOFTWARE_ENDSTOP_I
1416     #define MAX_SOFTWARE_ENDSTOP_J
1417     #define MAX_SOFTWARE_ENDSTOP_K
1418 #endif
1419
1420 #if EITHER(MIN_SOFTWARE_ENDSTOPS, MAX_SOFTWARE_ENDSTOPS)
1421     // #define SOFT_ENDSTOPS_MENU_ITEM // Enable/Disable software endstops
1422     // from the LCD
1423 #endif
1424
1425 /**
1426 * Filament Runout Sensors
1427 * Mechanical or opto endstops are used to check for the presence of
1428 * filament.
1429 *
1430 * IMPORTANT: Runout will only trigger if Marlin is aware that a print job
1431 * is running.
1432 * Marlin knows a print job is running when:
1433 *   1. Running a print job from media started with M24.
1434 *   2. The Print Job Timer has been started with M75.
1435 *   3. The heaters were turned on and PRINTJOB_TIMER_AUTOSTART is enabled.
1436 *
1437 * RAMPS-based boards use SERVO3_PIN for the first runout sensor.
1438 * For other boards you may need to define FIL_RUNOUT_PIN, FIL_RUNOUT2_PIN,
1439 * etc.
1440 */
1441 //#define FILAMENT_RUNOUT_SENSOR
1442 #if ENABLED(FILAMENT_RUNOUT_SENSOR)
1443     #define FIL_RUNOUT_ENABLED_DEFAULT true // Enable the sensor on startup.
1444     // Override with M412 followed by M500.
1445     #define NUM_RUNOUT_SENSORS 1           // Number of sensors, up to one
1446     // per extruder. Define a FIL_RUNOUT#_PIN for each.
1447
1448     #define FIL_RUNOUT_STATE      LOW      // Pin state indicating that
1449     // filament is NOT present.
1450     #define FIL_RUNOUT_PULLUP          // Use internal pullup for
1451     // filament runout pins.
1452     // #define FIL_RUNOUT_PULLDOWN        // Use internal pulldown for
1453     // filament runout pins.
1454     // #define WATCH_ALL_RUNOUT_SENSORS    // Execute runout script on any
1455     // triggering sensor, not only for the active extruder.
1456     // This is automatically enabled
1457     // for MIXING_EXTRUDERS.
1458
1459     // Override individually if the runout sensors vary
1460     // #define FIL_RUNOUT1_STATE LOW
1461     // #define FIL_RUNOUT1_PULLUP
1462     // #define FIL_RUNOUT1_PULLDOWN
1463
1464     // #define FIL_RUNOUT2_STATE LOW
1465     // #define FIL_RUNOUT2_PULLUP
1466     // #define FIL_RUNOUT2_PULLDOWN
1467
```

```
1457 //##define FIL_RUNOUT3_STATE LOW
1458 //##define FIL_RUNOUT3_PULLUP
1459 //##define FIL_RUNOUT3_PULLDOWN
1460
1461 //##define FIL_RUNOUT4_STATE LOW
1462 //##define FIL_RUNOUT4_PULLUP
1463 //##define FIL_RUNOUT4_PULLDOWN
1464
1465 //##define FIL_RUNOUT5_STATE LOW
1466 //##define FIL_RUNOUT5_PULLUP
1467 //##define FIL_RUNOUT5_PULLDOWN
1468
1469 //##define FIL_RUNOUT6_STATE LOW
1470 //##define FIL_RUNOUT6_PULLUP
1471 //##define FIL_RUNOUT6_PULLDOWN
1472
1473 //##define FIL_RUNOUT7_STATE LOW
1474 //##define FIL_RUNOUT7_PULLUP
1475 //##define FIL_RUNOUT7_PULLDOWN
1476
1477 //##define FIL_RUNOUT8_STATE LOW
1478 //##define FIL_RUNOUT8_PULLUP
1479 //##define FIL_RUNOUT8_PULLDOWN
1480
1481 // Commands to execute on filament runout.
1482 // With multiple runout sensors use the %c placeholder for the current
1483 // tool in commands (e.g., "M600 T%c")
1484 // NOTE: After 'M412 H1' the host handles filament runout and this script
1485 // does not apply.
1486 #define FILAMENT_RUNOUT_SCRIPT "M600"
1487
1488 // After a runout is detected, continue printing this length of filament
1489 // before executing the runout script. Useful for a sensor at the end of
1490 // a feed tube. Requires 4 bytes SRAM per sensor, plus 4 bytes overhead.
1491 //##define FILAMENT_RUNOUT_DISTANCE_MM 25
1492
1493 #ifdef FILAMENT_RUNOUT_DISTANCE_MM
1494     // Enable this option to use an encoder disc that toggles the runout pin
1495     // as the filament moves. (Be sure to set FILAMENT_RUNOUT_DISTANCE_MM
1496     // large enough to avoid false positives.)
1497     //##define FILAMENT_MOTION_SENSOR
1498 #endif
1499#endif
1500//=====
1501//=====
1502// @section calibrate
1503
1504/***
1505 * Choose one of the options below to enable G29 Bed Leveling. The
1506 * parameters
1507 * and behavior of G29 will change depending on your selection.
1508 *
1509 * If using a Probe for Z Homing, enable Z_SAFE_HOMING also!
1510 ..
```

```

1510 *
1511 * - AUTO_BED_LEVELING_3POINT
1512 *   Probe 3 arbitrary points on the bed (that aren't collinear)
1513 *   You specify the XY coordinates of all 3 points.
1514 *   The result is a single tilted plane. Best for a flat bed.
1515 *
1516 * - AUTO_BED_LEVELING_LINEAR
1517 *   Probe several points in a grid.
1518 *   You specify the rectangle and the density of sample points.
1519 *   The result is a single tilted plane. Best for a flat bed.
1520 *
1521 * - AUTO_BED_LEVELING_BILINEAR
1522 *   Probe several points in a grid.
1523 *   You specify the rectangle and the density of sample points.
1524 *   The result is a mesh, best for large or uneven beds.
1525 *
1526 * - AUTO_BED_LEVELING_UBL (Unified Bed Leveling)
1527 *   A comprehensive bed leveling system combining the features and benefits
1528 *   of other systems. UBL also includes integrated Mesh Generation, Mesh
1529 *   Validation and Mesh Editing systems.
1530 *
1531 * - MESH_BED_LEVELING
1532 *   Probe a grid manually
1533 *   The result is a mesh, suitable for large or uneven beds. (See
1534 *   BILINEAR.)
1535 *   For machines without a probe, Mesh Bed Leveling provides a method to
1536 *   perform
1537 *   leveling in steps so you can manually adjust the Z height at each grid-
1538 *   point.
1539 *   With an LCD controller the process is guided step-by-step.
1540 */
1541 // #define AUTO_BED_LEVELING_3POINT
1542 // #define AUTO_BED_LEVELING_LINEAR
1543 // #define AUTO_BED_LEVELING_BILINEAR
1544 // #define AUTO_BED_LEVELING_UBL
1545 #define MESH_BED_LEVELING

1546 /**
1547 * Normally G28 leaves leveling disabled on completion. Enable one of
1548 * these options to restore the prior leveling state or to always enable
1549 * leveling immediately after G28.
1550 */
1551 // #define RESTORE_LEVELING_AFTER_G28
1552 // #define ENABLE_LEVELING_AFTER_G28

1553 /**
1554 * Auto-leveling needs preheating
1555 */
1556 // #define PREHEAT_BEFORE_LEVELING
1557 #if ENABLED(PREHEAT_BEFORE_LEVELING)
1558   #define LEVELING_NOZZLE_TEMP 120    // (°C) Only applies to E0 at this time
1559   #define LEVELING_BED_TEMP      50
1560 #endif

1561 /**
1562 * Enable detailed logging of G28, G29, M48, etc.
1563 * Turn on with the command 'M111 S32'.
1564 * NOTE: Requires a lot of PROGMEM!
1565 */

```

```

1565 // #define DEBUG_LEVELING_FEATURE
1566
1567 #if ANY(MESH_BED_LEVELING, AUTO_BED_LEVELING_UBL, PROBE_MANUALLY)
1568     // Set a height for the start of manual adjustment
1569     #define MANUAL_PROBE_START_Z 1 // (mm) Comment out to use the last-
1570     measured height
1571 #endif
1572
1573 #if ANY(MESH_BED_LEVELING, AUTO_BED_LEVELING_BILINEAR,
1574 AUTO_BED_LEVELING_UBL)
1575     // Gradually reduce leveling correction until a set height is reached,
1576     // at which point movement will be level to the machine's XY plane.
1577     // The height can be set with M420 Z<height>
1578     #define ENABLE_LEVELING_FADE_HEIGHT
1579     #if ENABLED(ENABLE_LEVELING_FADE_HEIGHT)
1580         #define DEFAULT_LEVELING_FADE_HEIGHT 10.0 // (mm) Default fade height.
1581     #endif
1582
1583     // For Cartesian machines, instead of dividing moves on mesh boundaries,
1584     // split up moves into short segments like a Delta. This follows the
1585     // contours of the bed more closely than edge-to-edge straight moves.
1586     #define SEGMENT_LEVELED_MOVES
1587     #define LEVELED_SEGMENT_LENGTH 5.0 // (mm) Length of all segments (except
1588     the last one)
1589
1590 /**
1591 * Enable the G26 Mesh Validation Pattern tool.
1592 */
1593 #ifndef G26_MESH_VALIDATION
1594 #if ENABLED(G26_MESH_VALIDATION)
1595     #define MESH_TEST_NOZZLE_SIZE      0.4 // (mm) Diameter of primary
1596     nozzle.
1597     #define MESH_TEST_LAYER_HEIGHT    0.2 // (mm) Default layer height for
1598     G26.
1599     #define MESH_TEST_HOTEND_TEMP   205 // (°C) Default nozzle temperature
1600     for G26.
1601     #define MESH_TEST_BED_TEMP      60 // (°C) Default bed temperature
1602     for G26.
1603     #define G26_XY_FEEDRATE        20 // (mm/s) Feedrate for G26 XY
1604     moves.
1605     #define G26_XY_FEEDRATE_TRAVEL 100 // (mm/s) Feedrate for G26 XY
1606     travel moves.
1607     #define G26_RETRACT_MULTIPLIER  1.0 // G26 Q (retraction) used by
1608     default between mesh test elements.
1609 #endif
1610
1611#endif
1612
1613 #if EITHER(AUTO_BED_LEVELING_LINEAR, AUTO_BED_LEVELING_BILINEAR)
1614
1615     // Set the number of grid points per dimension.
1616     #define GRID_MAX_POINTS_X 3
1617     #define GRID_MAX_POINTS_Y GRID_MAX_POINTS_X
1618
1619     // Probe along the Y axis, advancing X after each column
1620     //#define PROBE_Y_FIRST
1621
1622 #if ENABLED(AUTO_BED_LEVELING_BILINEAR)

```

```
1614 // Beyond the probed grid, continue the implied tilt?  
1615 // Default is to maintain the height of the nearest edge.  
1616 //">#define EXTRAPOLATE_BEYOND_GRID  
1617  
1618 //  
1619 // Experimental Subdivision of the grid by Catmull-Rom method.  
1620 // Synthesizes intermediate points to produce a more detailed mesh.  
1621 //  
1622 //">#define ABL_BILINEAR_SUBDIVISION  
1623 #if ENABLED(ABL_BILINEAR_SUBDIVISION)  
1624     // Number of subdivisions between probe points  
1625     #define BILINEAR_SUBDIVISIONS 3  
1626 #endif  
1627  
1628 #endif  
1629  
1630 #elif ENABLED(AUTO_BED_LEVELING_UBL)  
1631  
1632 //=====  
1633 ====  
1633 //===== Unified Bed Leveling  
1634 =====  
1634 ====  
1635  
1636 //##define MESH_EDIT_GFX_OVERLAY // Display a graphics overlay while  
1636 editing the mesh  
1637  
1638 #define MESH_INSET 1 // Set Mesh bounds as an inset region of  
1638 the bed  
1639 #define GRID_MAX_POINTS_X 10 // Don't use more than 15 points per  
1639 axis, implementation limited.  
1640 #define GRID_MAX_POINTS_Y GRID_MAX_POINTS_X  
1641  
1642 //##define UBL_HILBERT_CURVE // Use Hilbert distribution for less  
1642 travel when probing multiple points  
1643  
1644 #define UBL_MESH_EDIT_MOVES_Z // Sophisticated users prefer no  
1644 movement of nozzle  
1645 #define UBL_SAVE_ACTIVE_ON_M500 // Save the currently active mesh in the  
1645 current slot on M500  
1646  
1647 //##define UBL_Z_RAISE_WHEN_OFF_MESH 2.5 // When the nozzle is off the  
1647 mesh, this value is used  
1648 // as the Z-Height correction  
1648 value.  
1649  
1650 //##define UBL_MESH_WIZARD // Run several commands in a row to get  
1650 a complete mesh  
1651  
1652 #elif ENABLED(MESH_BED_LEVELING)  
1653  
1654 //=====  
1654 ====  
1655 //===== Mesh  
1655 =====  
1656  
1656 ===
```

```

1657
1658 #define MESH_INSET 20           // Set Mesh bounds as an inset region of
1659   the bed
1660 #define GRID_MAX_POINTS_X 3    // Don't use more than 7 points per axis,
1661   implementation limited.
1662 #define GRID_MAX_POINTS_Y GRID_MAX_POINTS_X
1663
1664 //##define MESH_G28_REST_ORIGIN // After homing all axes ('G28' or 'G28
1665 XYZ') rest Z at Z_MIN_POS
1666#endif // BED_LEVELING
1667
1668 /**
1669 * Add a bed leveling sub-menu for ABL or MBL.
1670 * Include a guided procedure if manual probing is enabled.
1671 */
1672#define LCD_BED_LEVELING
1673
1674#if ENABLED(LCD_BED_LEVELING)
1675   #define MESH_EDIT_Z_STEP 0.025 // (mm) Step size while manually probing Z
1676   axis.
1677   #define LCD_PROBE_Z_RANGE 4    // (mm) Z Range centered on Z_MIN_POS for
1678   LCD Z adjustment
1679   #define MESH_EDIT_MENU        // Add a menu to edit mesh points
1680#endif
1681
1682// Add a menu item to move between bed corners for manual bed adjustment
1683#define LEVEL_BED_CORNERS
1684
1685#if ENABLED(LEVEL_BED_CORNERS)
1686   #define LEVEL_CORNERS_INSET_LFRB { 30, 30, 30, 30 } // (mm) Left, Front,
1687   Right, Back insets
1688   #define LEVEL_CORNERS_HEIGHT      0.0    // (mm) Z height of nozzle at
1689   leveling points
1690   #define LEVEL_CORNERS_Z_HOP       4.0    // (mm) Z height of nozzle between
1691   leveling points
1692   //##define LEVEL_CENTER_T00      // Move to the center after the
1693   last corner
1694   //##define LEVEL_CORNERS_USE_PROBE
1695   #if ENABLED(LEVEL_CORNERS_USE_PROBE)
1696     #define LEVEL_CORNERS_PROBE_TOLERANCE 0.1
1697     #define LEVEL_CORNERS_VERIFY_RAISED // After adjustment triggers the
1698     probe, re-probe to verify
1699     //##define LEVEL_CORNERS_AUDIO_FEEDBACK
1700  #endif
1701
1702 /**
1703 * Corner Leveling Order
1704 *
1705 * Set 2 or 4 points. When 2 points are given, the 3rd is the center of
1706 * the opposite edge.
1707 *
1708 * LF  Left-Front    RF  Right-Front
1709 * LB  Left-Back     RB  Right-Back
1710 *
1711 * Examples:
1712 *
1713 * Default          {LF,RF,LB,RF}          {LF,RF}          {LB,LF}
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2610
2611
2612
2613
2614
2615
2616
2617
2618
2619
2620
2621
2622
2623
2624
2625
2626
2627
2628
2629
2630
2631
2632
2633
2634
2635
2636
2637
2638
2639
2640
2641
2642
2643
2644
2645
2646
2647
2648
2649
2650
2651
2652
2653
2654
2655
2656
2657
2658
2659
2660
2661
2662
2663
2664
2665
2666
2667
2668
2669
2670
2671
2672
2673
2674
2675
2676
2677
2678
2679
2680
2681
2682
2683
2684
2685
2686
2687
2688
2689
2690
2691
2692
2693
2694
2695
2696
2697
2698
2699
2700
2701
2702
2703
2704
2705
2706
2707
2708
2709
2710
2711
2712
2713
2714
2715
2716
2717
2718
2719
2720
2721
2722
2723
2724
2725
2726
2727
2728
2729
2730
2731
2732
2733
2734
2735
2736
2737
2738
2739
2740
2741
2742
2743
2744
2745
2746
2747
2748
2749
2750
2751
2752
2753
2754
2755
2756
2757
2758
2759
2760
2761
2762
2763
2764
2765
2766
2767
2768
2769
2770
2771
2772
2773
2774
2775
2776
2777
2778
2779
2780
2781
2782
2783
2784
2785
2786
2787
2788
2789
2790
2791
2792
2793
2794
2795
2796
2797
2798
2799
2800
2801
2802
2803
2804
2805
2806
2807
2808
2809
2810
2811
2812
2813
2814
2815
2816
2817
2818
2819
2820
2821
2822
2823
2824
2825
2826
2827
2828
2829
2830
2831
2832
2833
2834
2835
2836
2837
2838
2839
2840
2841
2842
2843
2844
2845
2846
2847
2848
2849
2850
2851
2852
2853
2854
2855
2856
2857
2858
2859
2860
2861
2862
2863
2864
2865
2866
2867
2868
2869
2870
2871
2872
2873
2874
2875
2876
2877
2878
2879
2880
2881
2882
2883
2884
2885
2886
2887
2888
2889
2890
2891
2892
2893
2894
2895
2896
2897
2898
2899
2900
2901
2902
2903
2904
2905
2906
2907
2908
2909
2910
2911
2912
2913
2914
2915
2916
2917
2918
2919
2920
2921
2922
2923
2924
2925
2926
2927
2928
2929
2930
2931
2932
2933
2934
2935
2936
2937
2938
2939
2940
2941
2942
2943
2944
2945
2946
2947
2948
2949
2950
2951
2952
2953
2954
2955
2956
2957
2958
2959
2960
2961
2962
2963
2964
2965
2966
2967
2968
2969
2970
2971
2972
2973
2974
2975
2976
2977
2978
2979
2980
2981
2982
2983
2984
2985
2986
2987
2988
2989
2990
2991
2992
2993
2994
2995
2996
2997
2998
2999
2999

```

```

1704 *   LB ----- RB   LB ----- RB   LB ----- RB   LB ----- RB
1705 * | 4      3 | | 3      2 | | <3>    | | 1      |
1706 * |           | |           | |           | |           |
1707 * | 1      2 | | 1      4 | | 1      2 | | 2      |
1708 * LF ----- RF  LF ----- RF  LF ----- RF  LF ----- RF
1709 */
1710 #define LEVEL_CORNERS_LEVELING_ORDER { LF, RF, RB, LB }
1711#endif
1712
1713 /**
1714 * Commands to execute at the end of G29 probing.
1715 * Useful to retract or move the Z probe out of the way.
1716 */
1717 // #define Z_PROBE_END_SCRIPT "G1 Z10 F12000\nG1 X15 Y330\nG1 Z0.5\nG1 Z10"
1718
1719 // @section homing
1720
1721 // The center of the bed is at (X=0, Y=0)
1722 // #define BED_CENTER_AT_0_0
1723
1724 // Manually set the home position. Leave these undefined for automatic
1725 // settings.
1726 // For DELTA this is the top-center of the Cartesian print volume.
1727 // #define MANUAL_X_HOME_POS 0
1728 // #define MANUAL_Y_HOME_POS 0
1729 // #define MANUAL_Z_HOME_POS 0
1730 // #define MANUAL_I_HOME_POS 0
1731 // #define MANUAL_J_HOME_POS 0
1732 // #define MANUAL_K_HOME_POS 0
1733
1734 /**
1735 * Use "Z Safe Homing" to avoid homing with a Z probe outside the bed area.
1736 *
1737 * - Moves the Z probe (or nozzle) to a defined XY point before Z homing.
1738 * - Allows Z homing only when XY positions are known and trusted.
1739 * - If stepper drivers sleep, XY homing may be required again before Z
1740 homing.
1741 */
1742 // #define Z_SAFE_HOMING
1743
1744 #if ENABLED(Z_SAFE_HOMING)
1745     #define Z_SAFE_HOMING_X_POINT X_CENTER // X point for Z homing
1746     #define Z_SAFE_HOMING_Y_POINT Y_CENTER // Y point for Z homing
1747 #endif
1748
1749 // Homing speeds (mm/min)
1750 #define HOMING_FEEDRATE_MM_M { (50*60), (50*60), (4*60) }
1751
1752 // Validate that endstops are triggered on homing moves
1753 #define VALIDATE_HOMING_ENDSTOPS
1754
1755 // @section calibrate
1756
1757 /**
1758 * Bed Skew Compensation
1759 *
1760 * This feature corrects for misalignment in the XYZ axes.
1761 *
1762 * Take the following steps to get the bed skew in the XY plane:

```

```

1761 * 1. Print a test square (e.g., https://www.thingiverse.com/thing:2563185)
1762 * 2. For XY_DIAG_AC measure the diagonal A to C
1763 * 3. For XY_DIAG_BD measure the diagonal B to D
1764 * 4. For XY_SIDE_AD measure the edge A to D
1765 *
1766 * Marlin automatically computes skew factors from these measurements.
1767 * Skew factors may also be computed and set manually:
1768 *
1769 * - Compute AB      : SQRT(2*AC*AC+2*BD*BD-4*AD*AD)/2
1770 * - XY_SKEW_FACTOR : TAN(PI/2-ACOS((AC*AC-AB*AB-AD*AD)/(2*AB*AD)))
1771 *
1772 * If desired, follow the same procedure for XZ and YZ.
1773 * Use these diagrams for reference:
1774 *
1775 *      Y          Z          Z
1776 *      ^          ^          ^
1777 *      |          |          |
1778 *      |          /          /
1779 *      |          |          /
1780 *      |          A-----D   |          A-----D   |          A-----D
1781 *      +----->X      +----->X      +----->Y
1782 *      XY_SKEW_FACTOR    XZ_SKEW_FACTOR    YZ_SKEW_FACTOR
1783 */
1784 // #define SKEW_CORRECTION
1785
1786 #if ENABLED(SKEW_CORRECTION)
1787     // Input all length measurements here:
1788     #define XY_DIAG_AC 282.8427124746
1789     #define XY_DIAG_BD 282.8427124746
1790     #define XY_SIDE_AD 200
1791
1792     // Or, set the default skew factors directly here
1793     // to override the above measurements:
1794     #define XY_SKEW_FACTOR 0.0
1795
1796     // #define SKEW_CORRECTION_FOR_Z
1797     #if ENABLED(SKEW_CORRECTION_FOR_Z)
1798         #define XZ_DIAG_AC 282.8427124746
1799         #define XZ_DIAG_BD 282.8427124746
1800         #define YZ_DIAG_AC 282.8427124746
1801         #define YZ_DIAG_BD 282.8427124746
1802         #define YZ_SIDE_AD 200
1803         #define XZ_SKEW_FACTOR 0.0
1804         #define YZ_SKEW_FACTOR 0.0
1805     #endif
1806
1807     // Enable this option for M852 to set skew at runtime
1808     // #define SKEW_CORRECTION_GCODE
1809
1810 //=====
1811 ===
1812 //===== Additional Features
1813 ===
1814 // @section extras
1815

```

```
1816 /**
1817 * EEPROM
1818 *
1819 * Persistent storage to preserve configurable settings across reboots.
1820 *
1821 * M500 - Store settings to EEPROM.
1822 * M501 - Read settings from EEPROM. (i.e., Throw away unsaved changes)
1823 * M502 - Revert settings to "factory" defaults. (Follow with M500 to init
1824 * the EEPROM.)
1825 */
1826 #define EEPROM_SETTINGS // Persistent storage with M500 and M501
1827 //#define DISABLE_M503 // Saves ~2700 bytes of PROGMEM. Disable for
1828 release!
1829 #define EEPROM_CHITCHAT // Give feedback on EEPROM commands. Disable
1830 to save PROGMEM.
1831 #define EEPROM_BOOT_SILENT // Keep M503 quiet and only give errors during
1832 first load
1833 #if ENABLED(EEPROM_SETTINGS)
1834     #define EEPROM_AUTO_INIT // Init EEPROM automatically on any errors.
1835 #endif
1836
1837 /**
1838 * Host Keepalive
1839 */
1840 // When enabled Marlin will send a busy status message to the host
1841 // every couple of seconds when it can't accept commands.
1842 /**
1843 * HOST_KEEPALIVE_FEATURE // Disable this if your host doesn't
1844 like keepalive messages
1845 #define DEFAULT_KEEPALIVE_INTERVAL 2 // Number of seconds between "busy"
1846 messages. Set with M113.
1847 #define BUSY_WHILE_HEATING // Some hosts require "busy" messages
1848 even during heating
1849
1850 /**
1851 * G20/G21 Inch mode support
1852 */
1853 //#define INCH_MODE_SUPPORT
1854
1855 /**
1856 * M149 Set temperature units support
1857 */
1858 //#define TEMPERATURE_UNITS_SUPPORT
1859
1860 /**
1861 * @section temperature
1862 */
1863
1864 /**
1865 * Preheat Constants - Up to 5 are supported without changes
1866 */
1867
1868 #define PREHEAT_1_LABEL "PLA"
1869 #define PREHEAT_1_TEMP_HOTEND 180
1870 #define PREHEAT_1_TEMP_BED 70
1871 #define PREHEAT_1_TEMP_CHAMBER 45
1872 #define PREHEAT_1_FAN_SPEED 0 // Value from 0 to 255
1873
1874 #define PREHEAT_2_LABEL "ABS"
1875 #define PREHEAT_2_TEMP_HOTEND 240
1876 #define PREHEAT_2_TEMP_BED 110
1877 #define PREHEAT_2_TEMP_CHAMBER 50
```

```

1868 #define PREHEAT_2_FAN_SPEED      0 // Value from 0 to 255
1869
1870 #define PREHEAT_3_LABEL          "HDPE"
1871 #define PREHEAT_3_TEMP_HOTEND    250
1872 #define PREHEAT_3_TEMP_BED       110
1873 #define PREHEAT_3_TEMP_CHAMBER   70
1874 #define PREHEAT_3_FAN_SPEED      0 // Value from 0 to 255
1875
1876 /**
1877 * Nozzle Park
1878 *
1879 * Park the nozzle at the given XYZ position on idle or G27.
1880 *
1881 * The "P" parameter controls the action applied to the Z axis:
1882 *
1883 *     P0  (Default) If Z is below park Z raise the nozzle.
1884 *     P1  Raise the nozzle always to Z-park height.
1885 *     P2  Raise the nozzle by Z-park amount, limited to Z_MAX_POS.
1886 */
1887 //#define NOZZLE_PARK_FEATURE
1888
1889 #if ENABLED(NOZZLE_PARK_FEATURE)
1890     // Specify a park position as { X, Y, Z_raise }
1891     #define NOZZLE_PARK_POINT { (X_MIN_POS + 10), (Y_MAX_POS - 10), 20 }
1892     //#define NOZZLE_PARK_X_ONLY           // X move only is required to park
1893     //#define NOZZLE_PARK_Y_ONLY           // Y move only is required to park
1894     #define NOZZLE_PARK_Z_RAISE_MIN     2 // (mm) Always raise Z by at least
1895     this distance
1896     #define NOZZLE_PARK_XY_FEEDRATE 100 // (mm/s) X and Y axes feedrate
1897     (also used for delta Z axis)
1898     #define NOZZLE_PARK_Z_FEEDRATE     5 // (mm/s) Z axis feedrate (not used
1899     for delta printers)
1900 #endif
1901
1902 /**
1903 * Clean Nozzle Feature -- EXPERIMENTAL
1904 *
1905 * Adds the G12 command to perform a nozzle cleaning process.
1906 *
1907 * Parameters:
1908 *     P Pattern
1909 *     S Strokes / Repetitions
1910 *     T Triangles (P1 only)
1911 *
1912 * Patterns:
1913 *     P0 Straight line (default). This process requires a sponge type
1914 *     material
1915 *             at a fixed bed location. "S" specifies strokes (i.e. back-forth
1916 *     motions)
1917 *             between the start / end points.
1918 *
1919 *     P1 Zig-zag pattern between (X0, Y0) and (X1, Y1), "T" specifies the
1920 *     number of zig-zag triangles to do. "S" defines the number of
1921 *     strokes.
1922 *             Zig-zags are done in whichever is the narrower dimension.
1923 *             For example, "G12 P1 S1 T3" will execute:
1924 *
1925 *             -- ... . . . .. . . . .. . . . .. . . . .. . . . ..

```

```

1920 *
1921 *
1922 *
1923 *
1924 *
1925 *
1926 *
1927 *
1928 *
1929 * P2 Circular pattern with middle at NOZZLE_CLEAN_CIRCLE_MIDDLE.
1930 * "R" specifies the radius. "S" specifies the stroke count.
1931 * Before starting, the nozzle moves to NOZZLE_CLEAN_START_POINT.
1932 *
1933 * Caveats: The ending Z should be the same as starting Z.
1934 * Attention: EXPERIMENTAL. G-code arguments may change.
1935 */
1936 // #define NOZZLE_CLEAN_FEATURE
1937
1938 #if ENABLED(NOZZLE_CLEAN_FEATURE)
1939     // Default number of pattern repetitions
1940     #define NOZZLE_CLEAN_STROKES    12
1941
1942     // Default number of triangles
1943     #define NOZZLE_CLEAN_TRIANGLES  3
1944
1945     // Specify positions for each tool as { { X, Y, Z }, { X, Y, Z } }
1946     // Dual hotend system may use { { -20, (Y_BED_SIZE / 2), (Z_MIN_POS + 1) }
1947     // { 420, (Y_BED_SIZE / 2), (Z_MIN_POS + 1) } }
1948     #define NOZZLE_CLEAN_START_POINT { { 30, 30, (Z_MIN_POS + 1) } }
1949     #define NOZZLE_CLEAN_END_POINT   { { 100, 60, (Z_MIN_POS + 1) } }
1950
1951     // Circular pattern radius
1952     #define NOZZLE_CLEAN_CIRCLE_RADIUS 6.5
1953     // Circular pattern circle fragments number
1954     #define NOZZLE_CLEAN_CIRCLE_FN 10
1955     // Middle point of circle
1956     #define NOZZLE_CLEAN_CIRCLE_MIDDLE NOZZLE_CLEAN_START_POINT
1957
1958     // Move the nozzle to the initial position after cleaning
1959     #define NOZZLE_CLEAN_GOBACK
1960
1961     // For a purge/clean station that's always at the gantry height (thus no Z
1962     // move)
1963     // #define NOZZLE_CLEAN_NO_Z
1964
1965     // For a purge/clean station mounted on the X axis
1966     // #define NOZZLE_CLEAN_NO_Y
1967
1968     // Require a minimum hotend temperature for cleaning
1969     #define NOZZLE_CLEAN_MIN_TEMP 170
1970     // #define NOZZLE_CLEAN_HEATUP           // Heat up the nozzle instead of
1971     // skipping wipe
1972
1973     // Explicit wipe G-code script applies to a G12 with no arguments.
1974     // #define WIPE_SEQUENCE_COMMANDS "G1 X-17 Y25 Z10 F4000\nG1 Z1\nM114\nG1
X-17 Y25\nG1 X-17 Y95\nG1 X-17 Y25\nG1 X-17 Y95\nG1 X-17 Y25\nG1 X-17
Y95\nG1 X-17 Y25\nG1 X-17 Y95\nG1 X-17 Y25\nG1 X-17 Y95\nG1 X-17 Y25\nG1 X-
1975     // 17 Y95\nG1 Z15\nM400\nG0 X-10.0 Y-9.0"

```

```

1973 #endif
1974
1975 /**
1976 * Print Job Timer
1977 *
1978 * Automatically start and stop the print job timer on
1979 M104/M109/M140/M190/M141/M191.
1980 * The print job timer will only be stopped if the bed/chamber target temp
1981 is
1982 * below BED_MINTEMP/CHAMBER_MINTEMP.
1983 *
1984 * M104 (hotend, no wait) - high temp = none, low temp = stop
1985 timer
1986 * M109 (hotend, wait) - high temp = start timer, low temp = stop
1987 timer
1988 * M140 (bed, no wait) - high temp = none, low temp = stop
1989 timer
1990 * M190 (bed, wait) - high temp = start timer, low temp = none
1991 * M141 (chamber, no wait) - high temp = none, low temp = stop
1992 timer
1993 * M191 (chamber, wait) - high temp = start timer, low temp = none
1994 *
1995 * For M104/M109, high temp is anything over EXTRUDE_MINTEMP / 2.
1996 * For M140/M190, high temp is anything over BED_MINTEMP.
1997 * For M141/M191, high temp is anything over CHAMBER_MINTEMP.
1998 *
1999 * The timer can also be controlled with the following commands:
2000 *
2001 * M75 - Start the print job timer
2002 * M76 - Pause the print job timer
2003 * M77 - Stop the print job timer
2004 */
2005 #define PRINTJOB_TIMER_AUTOSTART
2006
2007 /**
2008 * Print Counter
2009 *
2010 * Track statistical data such as:
2011 *
2012 * - Total print jobs
2013 * - Total successful print jobs
2014 * - Total failed print jobs
2015 * - Total time printing
2016 *
2017 * View the current statistics with M78.
2018 */
2019 // #define PRINTCOUNTER
2020 #if ENABLED(PRINTCOUNTER)
2021   #define PRINTCOUNTER_SAVE_INTERVAL 60 // (minutes) EEPROM save interval
2022   during print
2023 #endif
2024
2025 /**
2026 * Password
2027 *
2028 * Set a numerical password for the printer which can be requested:
2029 *
2030 * - When the printer boots up

```

```

2024 * - Upon opening the 'Print from Media' Menu
2025 * - When SD printing is completed or aborted
2026 *
2027 * The following G-codes can be used:
2028 *
2029 * M510 - Lock Printer. Blocks all commands except M511.
2030 * M511 - Unlock Printer.
2031 * M512 - Set, Change and Remove Password.
2032 *
2033 * If you forget the password and get locked out you'll need to re-flash
2034 * the firmware with the feature disabled, reset EEPROM, and (optionally)
2035 * re-flash the firmware again with this feature enabled.
2036 */
2037 // #define PASSWORD_FEATURE
2038 #if ENABLED(PASSWORD_FEATURE)
2039     #define PASSWORD_LENGTH 4           // (#) Number of digits (1-9). 3
2040     or 4 is recommended
2041     #define PASSWORD_ON_STARTUP
2042     #define PASSWORD_UNLOCK_GCODE      // Unlock with the M511
2043     P<password> command. Disable to prevent brute-force attack.
2044     #define PASSWORD_CHANGE_GCODE      // Change the password with M512
2045     P<old> S<new>.
2046     // #define PASSWORD_ON_SD_PRINT_MENU    // This does not prevent gcodes
2047     from running
2048     // #define PASSWORD_AFTER_SD_PRINT_END
2049     // #define PASSWORD_AFTER_SD_PRINT_ABORT
2050     // #include "Configuration_Secure.h"    // External file with
2051     PASSWORD_DEFAULT_VALUE
2052 #endif
2053
2054 //=====
2055 ===
2056 //===== LCD and SD support
2057 =====
2058 ===
2059
2060 // @section lcd
2061
2062 /**
2063 * LCD LANGUAGE
2064 *
2065 * Select the language to display on the LCD. These languages are available:
2066 *
2067 * en, an, bg, ca, cz, da, de, el, el_CY, es, eu, fi, fr, gl, hr, hu, it,
2068 * jp_kana, ko_KR, nl, pl, pt, pt_BR, ro, ru, sk, sv, tr, uk, vi, zh_CN,
2069 * zh_TW
2070 *
2071 * :{ 'en':'English', 'an':'Aragonese', 'bg':'Bulgarian', 'ca':'Catalan',
2072 * 'cz':'Czech', 'da':'Danish', 'de':'German', 'el':'Greek (Greece)',
2073 * 'el_CY':'Greek (Cyprus)', 'es':'Spanish', 'eu':'Basque-Euskera',
2074 * 'fi':'Finnish', 'fr':'French', 'gl':'Galician', 'hr':'Croatian',
2075 * 'hu':'Hungarian', 'it':'Italian', 'jp_kana':'Japanese', 'ko_KR':'Korean
2076 * (South Korea)', 'nl':'Dutch', 'pl':'Polish', 'pt':'Portuguese',
2077 * 'pt_BR':'Portuguese (Brazilian)', 'ro':'Romanian', 'ru':'Russian',
2078 * 'sk':'Slovak', 'sv':'Swedish', 'tr':'Turkish', 'uk':'Ukrainian',
2079 * 'vi':'Vietnamese', 'zh_CN':'Chinese (Simplified)', 'zh_TW':'Chinese
2080 * (Traditional)' }
2081 */

```

```
2065 #define LCD_LANGUAGE en
2066
2067 /**
2068 * LCD Character Set
2069 *
2070 * Note: This option is NOT applicable to Graphical Displays.
2071 *
2072 * All character-based LCDs provide ASCII plus one of these
2073 * language extensions:
2074 *
2075 * - JAPANESE ... the most common
2076 * - WESTERN ... with more accented characters
2077 * - CYRILLIC ... for the Russian language
2078 *
2079 * To determine the language extension installed on your controller:
2080 *
2081 * - Compile and upload with LCD_LANGUAGE set to 'test'
2082 * - Click the controller to view the LCD menu
2083 * - The LCD will display Japanese, Western, or Cyrillic text
2084 *
2085 * See https://marlinfw.org/docs/development/lcd\_language.html
2086 *
2087 * :['JAPANESE', 'WESTERN', 'CYRILLIC']
2088 */
2089 #define DISPLAY_CHARSET_HD44780 WESTERN
2090
2091 /**
2092 * Info Screen Style (0:Classic, 1:Průša)
2093 *
2094 * :[0:'Classic', 1:'Průša']
2095 */
2096 #define LCD_INFO_SCREEN_STYLE 0
2097
2098 /**
2099 * SD CARD
2100 *
2101 * SD Card support is disabled by default. If your controller has an SD
2102 * slot,
2103 * you must uncomment the following option or it won't work.
2104 */
2104 #define SDSUPPORT
2105 // #define SD_CONNECTION_IS_ONBOARD
2106
2107 /**
2108 * SD CARD: ENABLE_CRC
2109 *
2110 * Use CRC checks and retries on the SD communication.
2111 */
2112 // #define SD_CHECK_AND_RETRY
2113
2114 /**
2115 * LCD Menu Items
2116 *
2117 * Disable all menus and only display the Status Screen, or
2118 * just remove some extraneous menu items to recover space.
2119 */
2120 // #define NO_LCD_MENUS
2121 // #define SLIM_LCD_MENUS
```

```
2122 //
2123 // Encoder Settings
2124 //
2125 // This option overrides the default number of encoder pulses needed to
2126 // produce one step. Should be increased for high-resolution encoders.
2127 //
2128 // #define ENCODER_PULSES_PER_STEP 4
2129 //
2130 //
2131 // Use this option to override the number of step signals required to
2132 // move between next/prev menu items.
2133 //
2134 // #define ENCODER_STEPS_PER_MENU_ITEM 1
2135 //
2136 /**
2137 * Encoder Direction Options
2138 *
2139 * Test your encoder's behavior first with both options disabled.
2140 *
2141 * Reversed Value Edit and Menu Nav? Enable REVERSE_ENCODER_DIRECTION.
2142 * Reversed Menu Navigation only? Enable REVERSE_MENU_DIRECTION.
2143 * Reversed Value Editing only? Enable BOTH options.
2144 */
2145 //
2146 //
2147 // This option reverses the encoder direction everywhere.
2148 //
2149 // Set this option if COUNTERCLOCKWISE causes values to DECREASE
2150 //
2151 // #define REVERSE_ENCODER_DIRECTION
2152 //
2153 //
2154 // This option reverses the encoder direction for navigating LCD menus.
2155 //
2156 // If COUNTERCLOCKWISE normally moves DOWN this makes it go UP.
2157 // If COUNTERCLOCKWISE normally moves UP this makes it go DOWN.
2158 //
2159 // #define REVERSE_MENU_DIRECTION
2160 //
2161 //
2162 // This option reverses the encoder direction for Select Screen.
2163 //
2164 // If COUNTERCLOCKWISE normally moves LEFT this makes it go RIGHT.
2165 // If COUNTERCLOCKWISE normally moves RIGHT this makes it go LEFT.
2166 //
2167 // #define REVERSE_SELECT_DIRECTION
2168 //
2169 //
2170 // Individual Axis Homing
2171 //
2172 // Add individual axis homing items (Home X, Home Y, and Home Z) to the LCD
2173 // menu.
2174 //
2175 #define INDIVIDUAL_AXIS_HOMING_MENU
2176 #define INDIVIDUAL_AXIS_HOMING_SUBMENU
2177 //
2178 //
2179 // <PFAKFR /RII77FR
```

```
// SPEAKER BUZZER
2180 // If you have a speaker that can produce tones, enable it here.
2181 // By default Marlin assumes you have a buzzer with a fixed frequency.
2182 //
2183 #define SPEAKER
2184 //
2185 //
2186 // The duration and frequency for the UI feedback sound.
2187 // Set these to 0 to disable audio feedback in the LCD menus.
2188 //
2189 // Note: Test audio output with the G-Code:
2190 // M300 S<frequency Hz> P<duration ms>
2191 //
2192 //#####
2193 //##define LCD_FEEDBACK_FREQUENCY_DURATION_MS 2
2194 //##define LCD_FEEDBACK_FREQUENCY_HZ 5000
2195 //
2196 //=====
2197 ===
2198 //===== LCD / Controller Selection
2199 =====
2200 //=====
2201 //
2202 // RepRapDiscount Smart Controller.
2203 // https://reprap.org/wiki/RepRapDiscount_Smart_Controller
2204 //
2205 // Note: Usually sold with a white PCB.
2206 //
2207 //##define REPRAP_DISCOUNT_SMART_CONTROLLER
2208 //
2209 //
2210 // GT2560 (YHCB2004) LCD Display
2211 //
2212 // Requires Testato, Koepel softwarewire library and
2213 // Andriy Golovnya's LiquidCrystal_AIP31068 library.
2214 //
2215 //##define YHCB2004
2216 //
2217 //
2218 // Original RADDS LCD Display+Encoder+SDCardReader
2219 // http://doku.radds.org/dokumentation/lcd-display/
2220 //
2221 //##define RADDS_DISPLAY
2222 //
2223 //
2224 // ULTIMAKER Controller.
2225 //
2226 //##define ULTIMAKERCONTROLLER
2227 //
2228 //
2229 // ULTIPANEL as seen on Thingiverse.
2230 //
2231 //##define ULTIPANEL
2232 //
2233 //
```

```
2234 // PanelOne from T3P3 (via RAMPS 1.4 AUX2/AUX3)
2235 // https://reprap.org/wiki/PanelOne
2236 //
2237 //#define PANEL_ONE
2238 //
2239 //
2240 // GADGETS3D G3D LCD/SD Controller
2241 // https://reprap.org/wiki/RAMPS_1.3/1.4_GADGETS3D_Shield_with_Panel
2242 //
2243 // Note: Usually sold with a blue PCB.
2244 //
2245 //#define G3D_PANEL
2246 //
2247 //
2248 // RigidBot Panel V1.0
2249 // http://www.inventapart.com/
2250 //
2251 //#define RIGIDBOT_PANEL
2252 //
2253 //
2254 // Makeboard 3D Printer Parts 3D Printer Mini Display 1602 Mini Controller
2255 // https://www.aliexpress.com/item/32765887917.html
2256 //
2257 //#define MAKEBOARD_MINI_2_LINE_DISPLAY_1602
2258 //
2259 //
2260 // ANET and Tronxy 20x4 Controller
2261 //
2262 //#define ZONESTAR_LCD          // Requires ADC_KEYPAD_PIN to be assigned
2263 // to an analog pin.
2264 //                                     // This LCD is known to be susceptible to
2265 // electrical interference           // which scrambles the display. Pressing
2266 // any button clears it up.           // This is a LCD2004 display with 5 analog
2267 // buttons.
2268 //
2269 // Generic 16x2, 16x4, 20x2, or 20x4 character-based LCD.
2270 //
2271 //#define ULTRA_LCD
2272 //=====
2273 ===
2274 //===== LCD / Controller Selection
2275 =====
2276 ===
2277 //
2278 // CONTROLLER TYPE: I2C
2279 //
2280 // Note: These controllers require the installation of Arduino's
2281 // LiquidCrystal_I2C
2282 // library. For more info: https://github.com/kiyoshigawa/LiquidCrystal_I2C
2283 //
```

```
2284 //  
2285 // Elefu RA Board Control Panel  
2286 // http://www.elefu.com/index.php?route=product/product&product_id=53  
2287 //  
2288 //#define RA_CONTROL_PANEL  
2289 //  
2290 // Sainsmart (YwRobot) LCD Displays  
2291 //  
2292 // These require F.Malpartida's LiquidCrystal_I2C library  
2293 // https://bitbucket.org/fmalpartida/new-liquidcrystal/wiki/Home  
2294 //  
2295 //#define LCD_SAINSMART_I2C_1602  
2296 //#define LCD_SAINSMART_I2C_2004  
2297 //  
2298 //  
2299 // Generic LCM1602 LCD adapter  
2300 //  
2301 //  
2302 //#define LCM1602  
2303 //  
2304 //  
2305 // PANELOLU2 LCD with status LEDs,  
2306 // separate encoder and click inputs.  
2307 //  
2308 // Note: This controller requires Arduino's LiquidTWI2 library v1.2.3 or  
later.  
2309 // For more info: https://github.com/lincomatic/LiquidTWI2  
2310 //  
2311 // Note: The PANELOLU2 encoder click input can either be directly connected  
to  
2312 // a pin (if BTN_ENC defined to != -1) or read through I2C (when BTN_ENC ==  
-1).  
2313 //  
2314 //#define LCD_I2C_PANELOLU2  
2315 //  
2316 //  
2317 // Panucatt VIKI LCD with status LEDs,  
2318 // integrated click & L/R/U/D buttons, separate encoder inputs.  
2319 //  
2320 //#define LCD_I2C_VIKI  
2321 //  
2322 //  
2323 // CONTROLLER TYPE: Shift register panels  
2324 //  
2325 //  
2326 //  
2327 // 2-wire Non-latching LCD SR from https://goo.gl/aJJ4sH  
2328 // LCD configuration: https://reprap.org/wiki/SAV_3D_LCD  
2329 //  
2330 //#define SAV_3DLCD  
2331 //  
2332 //  
2333 // 3-wire SR LCD with strobe using 74HC4094  
2334 // https://github.com/mikeshub/SailfishLCD  
2335 // Uses the code directly from Sailfish  
2336 //  
2337 //#define FF_INTERFACEBOARD  
2338 //
```

```
2339 //  
2340 // TFT GLCD Panel with Marlin UI  
2341 // Panel connected to main board by SPI or I2C interface.  
2342 // See https://github.com/Serhiy-K/TFTGLCDAdapter  
2343 //  
2344 //#define TFTGLCD_PANEL_SPI  
2345 //#define TFTGLCD_PANEL_I2C  
2346 //  
2347 //=====  
2348 ===== LCD / Controller Selection  
2349 ===== (Graphical LCDs)  
2350 //=====  
2351 //  
2352 //  
2353 // CONTROLLER TYPE: Graphical 128x64 (DOGM)  
2354 //  
2355 // IMPORTANT: The U8glib library is required for Graphical Display!  
2356 // https://github.com/olikraus/U8glib\_Arduino  
2357 //  
2358 // NOTE: If the LCD is unresponsive you may need to reverse the plugs.  
2359 //  
2360 //  
2361 //  
2362 // RepRapDiscount FULL GRAPHIC Smart Controller  
2363 // https://reprap.org/wiki/RepRapDiscount\_Full\_Graphic\_Smart\_Controller  
2364 //  
2365 //#define REPRAP_DISCOUNT_FULL_GRAPHIC_SMART_CONTROLLER  
2366 //  
2367 //  
2368 // K.3D Full Graphic Smart Controller  
2369 //  
2370 //#define K3D_FULL_GRAPHIC_SMART_CONTROLLER  
2371 //  
2372 //  
2373 // ReprapWorld Graphical LCD  
2374 // https://reprapworld.com/?products\_details&products\_id/1218  
2375 //  
2376 //#define REPRAPWORLD_GRAPHICAL_LCD  
2377 //  
2378 //  
2379 // Activate one of these if you have a Panucatt Devices  
2380 // Viki 2.0 or mini Viki with Graphic LCD  
2381 // https://www.panucatt.com  
2382 //  
2383 //#define VIKI2  
2384 //#define miniVIKI  
2385 //  
2386 //  
2387 // MakerLab Mini Panel with graphic  
2388 // controller and SD support - https://reprap.org/wiki/Mini\_panel  
2389 //  
2390 //#define MINIPANEL  
2391 //  
2392 //  
2393 // M4Kv2D Maker Board with graphic controller and SD support
```

```
// MKR3D MAKERBOARD WITH GRAPHIC CONTROLLER AND SD SUPPORT
2394 // https://reprap.org/wiki/MaKr3d_MaKrPanel
2395 //
2396 //#define MAKRPANEL
2397 //
2398 //
2399 // Adafruit ST7565 Full Graphic Controller.
2400 // https://github.com/eboston/Adafruit-ST7565-Full-Graphic-Controller/
2401 //
2402 //#define ELB_FULL_GRAPHIC_CONTROLLER
2403 //
2404 //
2405 // BQ LCD Smart Controller shipped by
2406 // default with the BQ Hephestos 2 and Witbox 2.
2407 //
2408 //#define BQ_LCD_SMART_CONTROLLER
2409 //
2410 //
2411 // Cartesio UI
2412 // http://mauk.cc/webshop/cartesio-shop/electronics/user-interface
2413 //
2414 //#define CARTESIO_UI
2415 //
2416 //
2417 // LCD for Melzi Card with Graphical LCD
2418 //
2419 //#define LCD_FOR_MELZI
2420 //
2421 //
2422 // Original Ulticontroller from Ultimaker 2 printer with SSD1309 I2C display
2423 // and encoder
2424 //
2425 // https://github.com/Ultimaker/Ultimaker2/tree/master/1249_Ulticontroller_Boar
2426 // d_(x1)
2427 //
2428 // MKS MINI12864 with graphic controller and SD support
2429 // https://reprap.org/wiki/MKS_MINI_12864
2430 //
2431 //#define MKS_MINI_12864
2432 //
2433 //
2434 // MKS MINI12864 V3 is an alias for FYSETC_MINI_12864_2_1. Type A/B.
2435 // NeoPixel RGB Backlight.
2436 //
2437 //#
2438 // MKS LCD12864A/B with graphic controller and SD support. Follows
2439 // MKS_MINI_12864 pinout.
2440 // https://www.aliexpress.com/item/33018110072.html
2441 //
2442 //#define MKS_LCD12864A
2443 //#define MKS_LCD12864B
2444 //
2445 //
2446 // FYSETC variant of the MINI12864 graphic controller with SD support
```

```
2447 // https://wiki.fysetc.com/Mini12864_Panel/
2448 //
2449 // #define FYSETC_MINI_12864_X_X      // Type C/D/E/F. No tunable RGB Backlight
2450 // by default
2451 // #define FYSETC_MINI_12864_1_2      // Type C/D/E/F. Simple RGB Backlight
2452 // (always on)
2453 // #define FYSETC_MINI_12864_2_0      // Type A/B. Discreet RGB Backlight
2454 // #define FYSETC_MINI_12864_2_1      // Type A/B. NeoPixel RGB Backlight
2455 // #define FYSETC_GENERIC_12864_1_1 // Larger display with basic ON/OFF
2456 // backlight.
2457 //
2458 // Factory display for Creality CR-10
2459 // https://www.aliexpress.com/item/32833148327.html
2460 //
2461 // This is RAMPS-compatible using a single 10-pin connector.
2462 // (For CR-10 owners who want to replace the Melzi Creality board but retain
2463 // the display)
2464 //
2465 #define CR10_STOCKDISPLAY
2466 //
2467 //
2468 // Ender-2 OEM display, a variant of the MKS_MINI_12864
2469 //
2470 // #define ENDER2_STOCKDISPLAY
2471 //
2472 //
2473 // ANET and Tronxy Graphical Controller
2474 // Anet 128x64 full graphics lcd with rotary encoder as used on Anet A6
2475 // A clone of the RepRapDiscount full graphics display but with
2476 // different pins/wiring (see pins_ANET_10.h). Enable one of these.
2477 //
2478 // #define ANET_FULL_GRAPHICS_LCD
2479 // #define ANET_FULL_GRAPHICS_LCD_ALT_WIRING
2480 //
2481 // AZSMZ 12864 LCD with SD
2482 // https://www.aliexpress.com/item/32837222770.html
2483 //
2484 // #define AZSMZ_12864
2485 //
2486 // Silvergate GLCD controller
2487 // https://github.com/android444/Silvergate
2488 //
2489 // #define SILVER_GATE_GLCD_CONTROLLER
2490 //
2491 //=====
2492 ===
2493 //===== OLED Displays
2494 =====
2495 //
2496 // SSD1306 OLED full graphics generic display
2497 //
2498 // #define I2C1_I2C_SSD1306
```

```
2490 // #define U8GLIB_SSD1306
2491
2500 //
2501 // SAV OLED LCD module support using either SSD1306 or SH1106 based LCD
2502 // modules
2503 //">#define SAV_3DGLCD
2504 #if ENABLED(SAV_3DGLCD)
2505     #define U8GLIB_SSD1306
2506     //">#define U8GLIB_SH1106
2507 #endif
2508
2509 //
2510 // TinyBoy2 128x64 OLED / Encoder Panel
2511 //
2512 //">#define OLED_PANEL_TINYBOY2
2513
2514 //
2515 // MKS OLED 1.3" 128x64 Full Graphics Controller
2516 // https://reprap.org/wiki/MKS_12864OLED
2517 //
2518 // Tiny, but very sharp OLED display
2519 //
2520 //">#define MKS_12864OLED          // Uses the SH1106 controller (default)
2521 //">#define MKS_12864OLED_SSD1306 // Uses the SSD1306 controller
2522
2523 //
2524 // Zonestar OLED 128x64 Full Graphics Controller
2525 //
2526 //">#define ZONESTAR_12864LCD      // Graphical (DOGM) with ST7920
2527 // controller
2528 //">#define ZONESTAR_12864OLED      // 1.3" OLED with SH1106 controller
2529 // (default)
2530 //">#define ZONESTAR_12864OLED_SSD1306 // 0.96" OLED with SSD1306 controller
2531
2532 //
2533 // Einstart S OLED SSD1306
2534 //
2535 //">#define U8GLIB_SH1106_EINSTART
2536
2537 //
2538 // Overlord OLED display/controller with i2c buzzer and LEDs
2539 //
2540 //">#define OVERLORD_OLED
2541
2542 // FYSETC OLED 2.42" 128x64 Full Graphics Controller with WS2812 RGB
2543 // Where to find : https://www.aliexpress.com/item/4000345255731.html
2544 //">#define FYSETC_242_OLED_12864 // Uses the SSD1309 controller
2545
2546 //
2547 // K.3D SSD1309 OLED 2.42" 128x64 Full Graphics Controller
2548 //
2549 //">#define K3D_242_OLED_CONTROLLER // Software SPI
2550
2551 //=====
===== Extensible UI Displays
=====
```

```
2552 //=====
2553 ===
2554 //
2555 // DGUS Touch Display with DWIN OS. (Choose one.)
2556 // ORIGIN : https://www.aliexpress.com/item/32993409517.html
2557 // FYSETC : https://www.aliexpress.com/item/32961471929.html
2558 // MKS     : https://www.aliexpress.com/item/1005002008179262.html
2559 //
2560 // Flash display with DGUS Displays for Marlin:
2561 //   - Format the SD card to FAT32 with an allocation size of 4kb.
2562 //   - Download files as specified for your type of display.
2563 //   - Plug the microSD card into the back of the display.
2564 //   - Boot the display and wait for the update to complete.
2565 //
2566 // ORIGIN (Marlin DWIN_SET)
2567 //   - Download https://github.com/coldtobi/Marlin_DGUS_Resources
2568 //   - Copy the downloaded DWIN_SET folder to the SD card.
2569 //
2570 // FYSETC (Supplier default)
2571 //   - Download https://github.com/FYSETC/FYSTLCD-2.0
2572 //   - Copy the downloaded SCREEN folder to the SD card.
2573 //
2574 // HIPRECY (Supplier default)
2575 //   - Download https://github.com/HiPrecy/Touch-Lcd-LE0
2576 //   - Copy the downloaded DWIN_SET folder to the SD card.
2577 //
2578 // MKS (MKS-H43) (Supplier default)
2579 //   - Download https://github.com/makerbase-mks/MKS-H43
2580 //   - Copy the downloaded DWIN_SET folder to the SD card.
2581 //
2582 // RELOADED (T5UID1)
2583 //   - Download https://github.com/Desuuuu/DGUS-reloaded/releases
2584 //   - Copy the downloaded DWIN_SET folder to the SD card.
2585 //
2586 // #define DGUS_LCD_UI_ORIGIN
2587 // #define DGUS_LCD_UI_FYSETC
2588 // #define DGUS_LCD_UI_HIPRECY
2589 // #define DGUS_LCD_UI_MKS
2590 // #define DGUS_LCD_UI_RELOADED
2591 #if ENABLED(DGUS_LCD_UI_MKS)
2592   #define USE_MKS_GREEN_UI
2593 #endif
2594 //
2595 //
2596 // Touch-screen LCD for Malyan M200/M300 printers
2597 //
2598 // #define MALYAN_LCD
2599 #if ENABLED(MALYAN_LCD)
2600   #define LCD_SERIAL_PORT 1 // Default is 1 for Malyan M200
2601 #endif
2602 //
2603 //
2604 // Touch UI for FTDI EVE (FT800/FT810) displays
2605 // See Configuration_adv.h for all configuration options.
2606 //
2607 // #define TOUCH_UI_FTDI_EVE
2608 //
2609 //
```

```
2609 //  
2610 // Touch-screen LCD for Anycubic printers  
2611 //  
2612 //##define ANYCUBIC_LCD_I3MEGA  
2613 //##define ANYCUBIC_LCD_CHIRON  
2614 #if EITHER(ANYCUBIC_LCD_I3MEGA, ANYCUBIC_LCD_CHIRON)  
    #define LCD_SERIAL_PORT 3 // Default is 3 for Anycubic  
    //##define ANYCUBIC_LCD_DEBUG  
#endif  
2618 //  
2619 // 320x240 Nextion 2.8" serial TFT Resistive Touch Screen NX3224T028  
2620 //  
2621 //##define NEXTION_TFT  
2622 #if ENABLED(NEXTION_TFT)  
    #define LCD_SERIAL_PORT 1 // Default is 1 for Nextion  
#endif  
2626 //  
2627 // Third-party or vendor-customized controller interfaces.  
2628 // Sources should be installed in 'src/lcd/extui'.  
2629 //  
2630 //##define EXTENSIBLE_UI  
2631 //  
2632 #if ENABLED(EXTENSIBLE_UI)  
    //##define EXTUI_LOCAL_BEEPER // Enables use of local Beeper pin with  
    external display  
#endif  
2636 //=====  
2637 ===  
2638 //===== Graphical TFTs  
2639 //=====  
2640 //  
2641 /**  
 * Specific TFT Model Presets. Enable one of the following options  
 * or enable TFT_GENERIC and set sub-options.  
 */  
2645 //  
2646 // 480x320, 3.5", SPI Display From MKS  
2647 // Normally used in MKS Robin Nano V2  
2648 //  
2649 //##define MKS_TS35_V2_0  
2650 //  
2652 //  
2653 // 320x240, 2.4", FSMC Display From MKS  
2654 // Normally used in MKS Robin Nano V1.2  
2655 //  
2656 //##define MKS_ROBIN_TFT24  
2657 //  
2658 //  
2659 // 320x240, 2.8", FSMC Display From MKS  
2660 // Normally used in MKS Robin Nano V1.2  
2661 //  
2662 //##define MKS_ROBIN_TFT28  
2663 //
```

```
2664 //  
2665 // 320x240, 3.2", FSMC Display From MKS  
2666 // Normally used in MKS Robin Nano V1.2  
2667 //  
2668 //#define MKS_ROBIN_TFT32  
2669 //  
2670 // 480x320, 3.5", FSMC Display From MKS  
2671 // Normally used in MKS Robin Nano V1.2  
2672 //  
2673 //#define MKS_ROBIN_TFT35  
2674 //  
2675 // 480x272, 4.3", FSMC Display From MKS  
2676 //  
2677 //#define MKS_ROBIN_TFT43  
2678 //  
2679 // 320x240, 3.2", FSMC Display From MKS  
2680 // Normally used in MKS Robin  
2681 //  
2682 //#define MKS_ROBIN_TFT_V1_1R  
2683 //  
2684 // 480x320, 3.5", FSMC Stock Display from TronxyXY  
2685 //  
2686 //#define TFT_TRONXY_X5SA  
2687 //  
2688 // 480x320, 3.5", FSMC Stock Display from ANYCUBIC  
2689 //  
2690 //#define ANYCUBIC_TFT35  
2691 //  
2692 // 320x240, 2.8", FSMC Stock Display from Longer/Alfawise  
2693 //  
2694 //#define LONGER_LK_TFT28  
2695 //  
2696 // 320x240, 2.8", FSMC Stock Display from ET4  
2697 //  
2698 //#define ANET_ET4_TFT28  
2699 //  
2700 // 480x320, 3.5", FSMC Stock Display from ET5  
2701 //  
2702 //#define ANET_ET5_TFT35  
2703 //  
2704 // 1024x600, 7", RGB Stock Display from BIQU-BX  
2705 //  
2706 //#define BIQU_BX_TFT70  
2707 //  
2708 // Generic TFT with detailed options  
2709 //  
2710 //#define TFT_GENERIC  
2711 //  
2712 // If enabled, will automatically select the best display  
2713 // based on the defined TFT_GENERIC  
2714 //  
2715 //#if ENABLED(TFT_GENERIC)  
2716 //  
2717 // Generic TFT with detailed options  
2718 //  
2719 //#define TFT_GENERIC  
2720 //#if ENABLED(TFT_GENERIC)  
2721 //  
2722 // If enabled, will automatically select the best display  
2723 // based on the defined TFT_GENERIC
```

```

2722 // :L "AUTO", "S1//SD", "S1//89", "S1//90", "ROTATE", "TFT19328",
2723 'ILI9341', 'ILI9488' ]
#define TFT_DRIVER AUTO

2724
2725 // Interface. Enable one of the following options:
2726 //#define TFT_INTERFACE_FSMC
2727 //#define TFT_INTERFACE_SPI
2728
2729 // TFT Resolution. Enable one of the following options:
2730 //#define TFT_RES_320x240
2731 //#define TFT_RES_480x272
2732 //#define TFT_RES_480x320
2733 //#define TFT_RES_1024x600
2734#endif
2735
2736 /**
2737 * TFT UI – User Interface Selection. Enable one of the following options:
2738 *
2739 * TFT_CLASSIC_UI – Emulated DOGM – 128x64 Upscaled
2740 * TFT_COLOR_UI – Marlin Default Menus, Touch Friendly, using full TFT
2741 capabilities
2742 * TFT_LVGL_UI – A Modern UI using LVGL
2743 *
2744 * For LVGL_UI also copy the 'assets' folder from the build directory to
2745 the
2746 * root of your SD card, together with the compiled firmware.
2747 */
2748 //#define TFT_CLASSIC_UI
2749 //#define TFT_COLOR_UI
2750 //#define TFT_LVGL_UI
2751
2752#if ENABLED(TFT_LVGL_UI)
2753    //#define MKS_WIFI_MODULE // MKS WiFi module
2754#endif
2755
2756 /**
2757 * TFT Rotation. Set to one of the following values:
2758 *
2759 * TFT_ROTATE_90, TFT_ROTATE_90_MIRROR_X, TFT_ROTATE_90_MIRROR_Y,
2760 * TFT_ROTATE_180, TFT_ROTATE_180_MIRROR_X, TFT_ROTATE_180_MIRROR_Y,
2761 * TFT_ROTATE_270, TFT_ROTATE_270_MIRROR_X, TFT_ROTATE_270_MIRROR_Y,
2762 * TFT_MIRROR_X, TFT_MIRROR_Y, TFT_NO_ROTATION
2763 */
2764 //#define TFT_ROTATION TFT_NO_ROTATION
2765
2766 //=====
2767 ===
2768 //===== Other Controllers
2769 =====
2770 ===
2771 // Ender-3 v2 OEM display. A DWIN display with Rotary Encoder.
2772 //
2773 //#define DWIN_CREALITY_LCD
2774 //
2775 // Ender-3 v2 OEM display. enhanced.

```

```

2775 // ...
2776 // #define DWIN_CREALITY_LCD_ENHANCED
2777 //
2778 // Ender-3 v2 OEM display with enhancements by Jacob Myers
2779 //
2780 // #define DWIN_CREALITY_LCD_JYERSUI
2781 //
2782 //
2783 // MarlinUI for Creality's DWIN display (and others)
2784 //
2785 // #define DWIN_MARLINUI_PORTRAIT
2786 // #define DWIN_MARLINUI_LANDSCAPE
2787 //
2788 //
2789 // Touch Screen Settings
2790 //
2791 // #define TOUCH_SCREEN
2792 #if ENABLED(TOUCH_SCREEN)
2793     #define BUTTON_DELAY_EDIT 50 // (ms) Button repeat delay for edit screens
2794     #define BUTTON_DELAY_MENU 250 // (ms) Button repeat delay for menus
2795
2796     // #define TOUCH_IDLE_SLEEP 300 // (secs) Turn off the TFT backlight if set
2797     // (5mn)
2798
2799     #define TOUCH_SCREEN_CALIBRATION
2800
2801     // #define TOUCH_CALIBRATION_X 12316
2802     // #define TOUCH_CALIBRATION_Y -8981
2803     // #define TOUCH_OFFSET_X      -43
2804     // #define TOUCH_OFFSET_Y      257
2805     // #define TOUCH_ORIENTATION TOUCH_LANDSCAPE
2806
2807     #if BOTH(TOUCH_SCREEN_CALIBRATION, EEPROM_SETTINGS)
2808         #define TOUCH_CALIBRATION_AUTO_SAVE // Auto save successful calibration
2809         values to EEPROM
2810     #endif
2811
2812     #if ENABLED(TFT_COLOR_UI)
2813         // #define SINGLE_TOUCH_NAVIGATION
2814     #endif
2815 #endif
2816 //
2817 // RepRapWorld REPRAPWORLD_KEYPAD v1.1
2818 //
2819 https://reprapworld.com/products/electronics/ramps/keypad_v1_0_fully_assembled/
2820 //
2821 // #define REPRAPWORLD_KEYPAD
2822 // #define REPRAPWORLD_KEYPAD_MOVE_STEP 10.0 // (mm) Distance to move per
2823 // key-press
2824 //=====
2825 ===
2826 //===== Extra Features
2827 =====
2828 //=====

```

```
===
2826 // @section extras
2827
2828 // Set number of user-controlled fans. Disable to use all board-defined
2829 // fans.
2830 // :[1,2,3,4,5,6,7,8]
2831 // #define NUM_M106_FANS 1
2832
2833 // Increase the FAN PWM frequency. Removes the PWM noise but increases
2834 // heating in the FET/Arduino
2835 // #define FAST_PWM_FAN
2836
2837 // Use software PWM to drive the fan, as for the heaters. This uses a very
2838 // low frequency
2839 // which is not as annoying as with the hardware PWM. On the other hand, if
2840 // this frequency
2841 // is too low, you should also increment SOFT_PWM_SCALE.
2842 // #define FAN_SOFT_PWM
2843
2844 // Incrementing this by 1 will double the software PWM frequency,
2845 // affecting heaters, and the fan if FAN_SOFT_PWM is enabled.
2846 // However, control resolution will be halved for each increment;
2847 // at zero value, there are 128 effective control positions.
2848 // :[0,1,2,3,4,5,6,7]
2849 #define SOFT_PWM_SCALE 0
2850
2851 // If SOFT_PWM_SCALE is set to a value higher than 0, dithering can
2852 // be used to mitigate the associated resolution loss. If enabled,
2853 // some of the PWM cycles are stretched so on average the desired
2854 // duty cycle is attained.
2855 // #define SOFT_PWM_DITHER
2856
2857 // Temperature status LEDs that display the hotend and bed temperature.
2858 // If all hotends, bed temperature, and target temperature are under 54C
2859 // then the BLUE led is on. Otherwise the RED led is on. (1C hysteresis)
2860 // #define TEMP_STAT_LEDS
2861
2862 // Support for the BaricUDA Paste Extruder
2863 // #define BARICUDA
2864
2865 // Support for BlinkM/CyzRgb
2866 // #define BLINKM
2867
2868 // Support for PCA9632 PWM LED driver
2869 // #define PCA9632
2870
2871 /**
2872 * RGB LED / LED Strip Control
2873 *
2874 * Enable support for an RGB LED connected to 5V digital pins, or
2875 * an RGB Strip connected to MOSFETs controlled by digital pins.
2876 *
2877 * Adds the M150 command to set the LED (or LED strip) color.
2878 * If pins are PWM capable (e.g., 4, 5, 6, 11) then a range of
2879 * luminance values can be set from 0 to 255.
```

```

2880 * For NeoPixel LED an overall brightness parameter is also available.
2881 *
2882 * *** CAUTION ***
2883 * LED Strips require a MOSFET Chip between PWM lines and LEDs,
2884 * as the Arduino cannot handle the current the LEDs will require.
2885 * Failure to follow this precaution can destroy your Arduino!
2886 * NOTE: A separate 5V power supply is required! The NeoPixel LED needs
2887 * more current than the Arduino 5V linear regulator can produce.
2888 * *** CAUTION ***
2889 *
2900 * LED Type. Enable only one of the following two options.
2901 */
2902 //#define RGB_LED
2903 //#define RGBW_LED
2904
2905 #if EITHER(RGB_LED, RGBW_LED)
2906     //#define RGB_LED_R_PIN 34
2907     //#define RGB_LED_G_PIN 43
2908     //#define RGB_LED_B_PIN 35
2909     //#define RGB_LED_W_PIN -1
2910#endif
2911
2912 // Support for Adafruit NeoPixel LED driver
2913 //#define NEOPIXEL_LED
2914 #if ENABLED(NEOPIXEL_LED)
2915     #define NEOPIXEL_TYPE    NEO_GRBW // NEO_GRBW / NEO_GRB – four/three
channel driver type (defined in Adafruit_NeoPixel.h)
2916     //#define NEOPIXEL_PIN      4    // LED driving pin
2917     //#define NEOPIXEL2_TYPE NEOPIXEL_TYPE
2918     //#define NEOPIXEL2_PIN      5
2919     #define NEOPIXEL_PIXELS 30        // Number of LEDs in the strip. (Longest
strip when NEOPIXEL2_SEPARATE is disabled.)
2920     #define NEOPIXEL_IS_SEQUENTIAL // Sequential display for temperature
change – LED by LED. Disable to change all LEDs at once.
2921     #define NEOPIXEL_BRIGHTNESS 127 // Initial brightness (0–255)
2922     //#define NEOPIXEL_STARTUP_TEST // Cycle through colors at startup
2923
2924 // Support for second Adafruit NeoPixel LED driver controlled with M150 S1
2925 ...
2926
2927 // #define NEOPIXEL2_SEPARATE
2928 #if ENABLED(NEOPIXEL2_SEPARATE)
2929     #define NEOPIXEL2_PIXELS    15 // Number of LEDs in the second strip
2930     #define NEOPIXEL2_BRIGHTNESS 127 // Initial brightness (0–255)
2931     #define NEOPIXEL2_STARTUP_TEST // Cycle through colors at startup
2932 #else
2933     // #define NEOPIXEL2_INSERIES // Default behavior is NeoPixel 2 in
parallel
2934 #endif
2935
2936 // Use some of the NeoPixel LEDs for static (background) lighting
2937     //#define NEOPIXEL_BKGD_INDEX_FIRST 0           // Index of the first
background LED
2938     //#define NEOPIXEL_BKGD_INDEX_LAST   5           // Index of the last
background LED
2939     //#define NEOPIXEL_BKGD_COLOR { 255, 255, 255, 0 } // R, G, B, W
2940     //#define NEOPIXEL_BKGD_ALWAYS_ON          // Keep the backlight
on when other NeoPixels are off
2941#endif

```

```
2930
2931 /**
2932 * Printer Event LEDs
2933 *
2934 * During printing, the LEDs will reflect the printer status:
2935 *
2936 * - Gradually change from blue to violet as the heated bed gets to target
2937 * temp
2938 * - Gradually change from violet to red as the hotend gets to temperature
2939 * - Change to white to illuminate work surface
2940 * - Change to green once print has finished
2941 * - Turn off after the print has finished and the user has pushed a button
2942 */
2943 #if ANY(BLINKM, RGB_LED, RGBW_LED, PCA9632, PCA9533, NEOPIXEL_LED)
2944     #define PRINTER_EVENT_LEDS
2945 #endif
2946
2947 /**
2948 * Number of servos
2949 *
2950 * For some servo-related options NUM_SERVOS will be set automatically.
2951 * Set this manually if there are extra servos needing manual control.
2952 * Set to 0 to turn off servo support.
2953 */
2954 // #define NUM_SERVOS 3 // Note: Servo index starts with 0 for M280-M282
2955 // commands
2956 // (ms) Delay before the next move will start, to give the servo time to
2957 // reach its target angle.
2958 // 300ms is a good value but you can try less delay.
2959 // If the servo can't reach the requested position, increase it.
2960 #define SERVO_DELAY { 300 }
2961
2962 // Only power servos during movement, otherwise leave off to prevent jitter
2963 // #define DEACTIVATE_SERVOS_AFTER_MOVE
2964
2965 // Edit servo angles with M281 and save to EEPROM with M500
2966 // #define EDITABLE_SERVO_ANGLES
2967
2968 // Disable servo with M282 to reduce power consumption, noise, and heat when
2969 // not in use
2970 // #define SERVO_DETACH_GCODE
```

I CONFIGURATION ADVANCED

```
1  /**
2   * Marlin 3D Printer Firmware
3   * Copyright (c) 2020 MarlinFirmware
4   * [https://github.com/MarlinFirmware/Marlin]
5   *
6   * Based on Sprinter and grbl.
7   * Copyright (c) 2011 Camiel Gubbels / Erik van der Zalm
8   *
9   * This program is free software: you can redistribute it and/or modify
10  * it under the terms of the GNU General Public License as published by
11  * the Free Software Foundation, either version 3 of the License, or
12  * (at your option) any later version.
13  *
14  * This program is distributed in the hope that it will be useful,
15  * but WITHOUT ANY WARRANTY; without even the implied warranty of
16  * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
17  * GNU General Public License for more details.
18  *
19  * You should have received a copy of the GNU General Public License
20  * along with this program. If not, see <https://www.gnu.org/licenses/>.
21  */
22 #pragma once
23
24 /**
25  * Configuration_adv.h
26  *
27  * Advanced settings.
28  * Only change these if you know exactly what you're doing.
29  * Some of these settings can damage your printer if improperly set!
30  *
31  * Basic settings can be found in Configuration.h
32  */
33 #define CONFIGURATION_ADV_H_VERSION 02000902
34
35 //=====
36 =
37 //===== Thermal Settings
38 =
39 // @section temperature
40
41 /**
42  * Thermocouple sensors are quite sensitive to noise. Any noise induced in
43  * the sensor wires, such as by stepper motor wires run in parallel to them,
44  * may result in the thermocouple sensor reporting spurious errors. This
45  * value is the number of errors which can occur in a row before the error
46  * is reported. This allows us to ignore intermittent error conditions
47  * while
48  * still detecting an actual failure, which should result in a continuous
49  * stream of errors from the sensor.
50  *
51  * Set this value to 0 to fail on the first error to occur.
52  */
53 #define THERMOCOUPLE_MAX_ERRORS 15
54
55 //
```

```
54 // Custom Thermistor 1000 parameters
55 //
56 #if TEMP_SENSOR_0 == 1000
57     #define HOTEND0_PULLUP_RESISTOR_OHMS 4700      // Pullup resistor
58     #define HOTEND0_RESISTANCE_25C_OHMS 100000      // Resistance at 25C
59     #define HOTEND0_BETA                 3950      // Beta value
60 #endif
61
62 #if TEMP_SENSOR_1 == 1000
63     #define HOTEND1_PULLUP_RESISTOR_OHMS 4700      // Pullup resistor
64     #define HOTEND1_RESISTANCE_25C_OHMS 100000      // Resistance at 25C
65     #define HOTEND1_BETA                 3950      // Beta value
66 #endif
67
68 #if TEMP_SENSOR_2 == 1000
69     #define HOTEND2_PULLUP_RESISTOR_OHMS 4700      // Pullup resistor
70     #define HOTEND2_RESISTANCE_25C_OHMS 100000      // Resistance at 25C
71     #define HOTEND2_BETA                 3950      // Beta value
72 #endif
73
74 #if TEMP_SENSOR_3 == 1000
75     #define HOTEND3_PULLUP_RESISTOR_OHMS 4700      // Pullup resistor
76     #define HOTEND3_RESISTANCE_25C_OHMS 100000      // Resistance at 25C
77     #define HOTEND3_BETA                 3950      // Beta value
78 #endif
79
80 #if TEMP_SENSOR_4 == 1000
81     #define HOTEND4_PULLUP_RESISTOR_OHMS 4700      // Pullup resistor
82     #define HOTEND4_RESISTANCE_25C_OHMS 100000      // Resistance at 25C
83     #define HOTEND4_BETA                 3950      // Beta value
84 #endif
85
86 #if TEMP_SENSOR_5 == 1000
87     #define HOTEND5_PULLUP_RESISTOR_OHMS 4700      // Pullup resistor
88     #define HOTEND5_RESISTANCE_25C_OHMS 100000      // Resistance at 25C
89     #define HOTEND5_BETA                 3950      // Beta value
90 #endif
91
92 #if TEMP_SENSOR_6 == 1000
93     #define HOTEND6_PULLUP_RESISTOR_OHMS 4700      // Pullup resistor
94     #define HOTEND6_RESISTANCE_25C_OHMS 100000      // Resistance at 25C
95     #define HOTEND6_BETA                 3950      // Beta value
96 #endif
97
98 #if TEMP_SENSOR_7 == 1000
99     #define HOTEND7_PULLUP_RESISTOR_OHMS 4700      // Pullup resistor
100    #define HOTEND7_RESISTANCE_25C_OHMS 100000      // Resistance at 25C
101    #define HOTEND7_BETA                 3950      // Beta value
102 #endif
103
104 #if TEMP_SENSOR_BED == 1000
105     #define BED_PULLUP_RESISTOR_OHMS      4700      // Pullup resistor
106     #define BED_RESISTANCE_25C_OHMS     100000      // Resistance at 25C
107     #define BED_BETA                   3950      // Beta value
108 #endif
109
110 #if TEMP_SENSOR_CHAMBER == 1000
111     #define CHAMBER_PULLUP_RESISTOR_OHMS 4700      // Pullup resistor
112     #define CHAMBER_RESISTANCE_25C_OHMS 100000      // Resistance at 25C
```

```

113 #define CHAMBER_BETA           3950 // Beta value
114 #endif
115
116 #if TEMP_SENSOR_COOLER == 1000
117   #define COOLER_PULLUP_RESISTOR_OHMS 4700 // Pullup resistor
118   #define COOLER_RESISTANCE_25C_OHMS 100000 // Resistance at 25C
119   #define COOLER_BETA                 3950 // Beta value
120 #endif
121
122 #if TEMP_SENSOR_PROBE == 1000
123   #define PROBE_PULLUP_RESISTOR_OHMS 4700 // Pullup resistor
124   #define PROBE_RESISTANCE_25C_OHMS 100000 // Resistance at 25C
125   #define PROBE_BETA                 3950 // Beta value
126 #endif
127
128 #if TEMP_SENSOR_BOARD == 1000
129   #define BOARD_PULLUP_RESISTOR_OHMS 4700 // Pullup resistor
130   #define BOARD_RESISTANCE_25C_OHMS 100000 // Resistance at 25C
131   #define BOARD_BETA                 3950 // Beta value
132 #endif
133
134 #if TEMP_SENSOR_REDUNDANT == 1000
135   #define REDUNDANT_PULLUP_RESISTOR_OHMS 4700 // Pullup resistor
136   #define REDUNDANT_RESISTANCE_25C_OHMS 100000 // Resistance at 25C
137   #define REDUNDANT_BETA                 3950 // Beta value
138 #endif
139
140 /**
141 * Configuration options for MAX Thermocouples (-2, -3, -5).
142 * FORCE_HW_SPI: Ignore SCK/MOSI/MISO pins and just use the CS pin &
143 * default SPI bus.
144 * MAX31865_WIRES: Set the number of wires for the probe connected to a
145 * MAX31865 board, 2-4. Default: 2
146 * MAX31865_50HZ: Enable 50Hz filter instead of the default 60Hz.
147 */
148 // #define TEMP_SENSOR_FORCE_HW_SPI
149 // #define MAX31865_SENSOR_WIRES_0 2
150 // #define MAX31865_SENSOR_WIRES_1 2
151 // #define MAX31865_50HZ_FILTER
152
153 /**
154 * Hephestos 2 24V heated bed upgrade kit.
155 * https://store.bq.com/en/heated-bed-kit-hephestos2
156 */
157 // #define HEPHESTOS2_HEATED_BED_KIT
158 #if ENABLED(HEPHESTOS2_HEATED_BED_KIT)
159   #undef TEMP_SENSOR_BED
160   #define TEMP_SENSOR_BED 70
161   #define HEATER_BED_INVERTING true
162 #endif
163
164 /**
165 * Heated Bed Bang-Bang options
166 */
167 #if DISABLED(PIDTEMPBED)
168   #define BED_CHECK_INTERVAL 5000 // (ms) Interval between checks in bang-
169   // bang control
170   #if ENABLED(BED_LIMIT_SWITCHING)

```

```

168     #define BED_HYSTERESIS 2          // (°C) Only set the relevant heater
169     state when ABS(T-target) > BED_HYSTERESIS
170 #endif
171
172 //
173 // Heated Chamber options
174 //
175 #if DISABLED(PIDTEMPCHAMBER)
176     #define CHAMBER_CHECK_INTERVAL 5000    // (ms) Interval between checks in
177     bang-bang control
178     #if ENABLED(CHAMBER_LIMIT_SWITCHING)
179         #define CHAMBER_HYSTERESIS 2          // (°C) Only set the relevant heater
180         state when ABS(T-target) > CHAMBER_HYSTERESIS
181     #endif
182 #endif
183
184 #if TEMP_SENSOR_CHAMBER
185     #define HEATER_CHAMBER_PIN      HEATER_1_PIN    // Required heater on/off
186     pin (example: SKR 1.4 Turbo HE1 plug)
187     //##define HEATER_CHAMBER_INVERTING false
188     //##define FAN1_PIN           -1    // Remove the fan signal on pin
189     P2_04 (example: SKR 1.4 Turbo HE1 plug)
190
191     //##define CHAMBER_FAN           // Enable a fan on the chamber
192     #if ENABLED(CHAMBER_FAN)
193         #define CHAMBER_FAN_MODE 2        // Fan control mode: 0=Static;
194         1=Linear increase when temp is higher than target; 2=V-shaped curve;
195         3=similar to 1 but fan is always on.
196         #if CHAMBER_FAN_MODE == 0
197             #define CHAMBER_FAN_BASE 255   // Chamber fan PWM (0-255)
198         #elif CHAMBER_FAN_MODE == 1
199             #define CHAMBER_FAN_BASE 128   // Base chamber fan PWM (0-255); turns
200             on when chamber temperature is above the target
201             #define CHAMBER_FAN_FACTOR 25  // PWM increase per °C above target
202         #elif CHAMBER_FAN_MODE == 2
203             #define CHAMBER_FAN_BASE 128   // Minimum chamber fan PWM (0-255)
204             #define CHAMBER_FAN_FACTOR 25  // PWM increase per °C difference from
205             target
206         #elif CHAMBER_FAN_MODE == 3
207             #define CHAMBER_FAN_BASE 128   // Base chamber fan PWM (0-255)
208             #define CHAMBER_FAN_FACTOR 25  // PWM increase per °C above target
209         #endif
210     #endif
211
212     //##define CHAMBER_VENT           // Enable a servo-controlled vent on
213     the chamber
214     #if ENABLED(CHAMBER_VENT)
215         #define CHAMBER_VENT_SERVO_NR 1 // Index of the vent servo
216         #define HIGH_EXCESS_HEAT_LIMIT 5 // How much above target temp to
217         consider there is excess heat in the chamber
218         #define LOW_EXCESS_HEAT_LIMIT 3
219         #define MIN_COOLING_SLOPE_TIME_CHAMBER_VENT 20
220         #define MIN_COOLING_SLOPE_DEG_CHAMBER_VENT 1.5
221     #endif
222 #endif
223
224 //
225 // Laser cooler options

```

```

--> // Laser cooler options
216 // 
217 #if TEMP_SENSOR_COOLER
218     #define COOLER_MINTEMP          8 // (°C)
219     #define COOLER_MAXTEMP          26 // (°C)
220     #define COOLER_DEFAULT_TEMP     16 // (°C)
221     #define TEMP_COOLER_HYSTERESIS  1 // (°C) Temperature proximity
222         considered "close enough" to the target
223     #define COOLER_PIN              8 // Laser cooler on/off pin used to
224         control power to the cooling element (e.g., TEC, External chiller via relay)
225     #define COOLER_INVERTING        false
226     #define TEMP_COOLER_PIN         15 // Laser/Cooler temperature sensor
227         pin. ADC is required.
228     #define COOLER_FAN               // Enable a fan on the cooler, Fan#
229         0,1,2,3 etc.
230     #define COOLER_FAN_INDEX        0 // FAN number 0, 1, 2 etc. e.g.
231     #if ENABLED(COOLER_FAN)
232         #define COOLER_FAN_BASE      100 // Base Cooler fan PWM (0-255); turns
233             on when Cooler temperature is above the target
234         #define COOLER_FAN_FACTOR    25 // PWM increase per °C above target
235     #endif
236 #endif

237 // 
238 // Motherboard Sensor options
239 //
240 #if TEMP_SENSOR_BOARD
241     #define THERMAL_PROTECTION_BOARD // Halt the printer if the board sensor
242         leaves the temp range below.
243     #define BOARD_MINTEMP          8 // (°C)
244     #define BOARD_MAXTEMP          70 // (°C)
245     #ifndef TEMP_BOARD_PIN
246         // #define TEMP_BOARD_PIN -1 // Board temp sensor pin, if not set in
247             pins file.
248     #endif
249 #endif

250 // 
251 // Laser Coolant Flow Meter
252 //
253 // #define LASER_COOLANT_FLOW_METER
254 #if ENABLED(LASER_COOLANT_FLOW_METER)
255     #define FLOWMETER_PIN           20 // Requires an external interrupt-
256         enabled pin (e.g., RAMPS 2,3,18,19,20,21)
257     #define FLOWMETER_PPL            5880 // (pulses/liter) Flow meter pulses-per-
258         liter on the input pin
259     #define FLOWMETER_INTERVAL      1000 // (ms) Flow rate calculation interval
260         in milliseconds
261     #define FLOWMETER_SAFETY        // Prevent running the laser without the
262         minimum flow rate set below
263     #if ENABLED(FLOWMETER_SAFETY)
264         #define FLOWMETER_MIN_LITERS_PER_MINUTE 1.5 // (liters/min) Minimum flow
265             required when enabled
266     #endif
267 #endif

268 /**
269  * Thermal Protection provides additional protection to your printer from
270  * damage

```

```

261 * and fire. Marlin always includes safe min and max temperature ranges
262 which
263 * protect against a broken or disconnected thermistor wire.
264 *
265 * The issue: If a thermistor falls out, it will report the much lower
266 * temperature of the air in the room, and the the firmware will keep
267 * the heater on.
268 *
269 * The solution: Once the temperature reaches the target, start observing.
270 * If the temperature stays too far below the target (hysteresis) for too
271 * long (period), the firmware will halt the machine as a safety precaution.
272 *
273 * If you get false positives for "Thermal Runaway", increase
274 * THERMAL_PROTECTION_HYSTERESIS and/or THERMAL_PROTECTION_PERIOD
275 */
276 #if ENABLED(THERMAL_PROTECTION_HOTENDS)
277     #define THERMAL_PROTECTION_PERIOD 40          // Seconds
278     #define THERMAL_PROTECTION_HYSTERESIS 4         // Degrees Celsius
279
280     //##define ADAPTIVE_FAN_SLOWING                // Slow part cooling fan if
281     // temperature drops
282     #if BOTH(ADAPTIVE_FAN_SLOWING, PIDTEMP)
283         //##define NO_FAN_SLOWING_IN_PID_TUNING      // Don't slow fan speed during
284     M303
285     #endif
286
287     /**
288     * Whenever an M104, M109, or M303 increases the target temperature, the
289     * firmware will wait for the WATCH_TEMP_PERIOD to expire. If the
290     * temperature
291     * hasn't increased by WATCH_TEMP_INCREASE degrees, the machine is halted
292     * and
293     * requires a hard reset. This test restarts with any M104/M109/M303, but
294     * only
295     * if the current temperature is far enough below the target for a
296     * reliable
297     * test.
298     *
299     * If you get false positives for "Heating failed", increase
300     WATCH_TEMP_PERIOD
301     * and/or decrease WATCH_TEMP_INCREASE. WATCH_TEMP_INCREASE should not be
302     set
303     * below 2.
304     */
305     #define WATCH_TEMP_PERIOD 20                  // Seconds
306     #define WATCH_TEMP_INCREASE 2                 // Degrees Celsius
307 #endif
308
309 /**
310 * Thermal Protection parameters for the bed are just as above for hotends.
311 */
312 #if ENABLED(THERMAL_PROTECTION_BED)
313     #define THERMAL_PROTECTION_BED_PERIOD        20 // Seconds
314     #define THERMAL_PROTECTION_BED_HYSTERESIS    2 // Degrees Celsius
315
316 /**
317 * As described above, except for the bed (M140/M190/M303).
318 */
319 #define WATCH_RFD_TFMP_PFRTOA 60 // Seconds

```

```

311 // seconds
312 #define WATCH_BED_TEMP_INCREASE 2 // Degrees Celsius
313 #endif

314 /**
315 * Thermal Protection parameters for the heated chamber.
316 */
317 #if ENABLED(THERMAL_PROTECTION_CHAMBER)
318 #define THERMAL_PROTECTION_CHAMBER_PERIOD 240 // Seconds
319 #define THERMAL_PROTECTION_CHAMBER_HYSTERESIS 1 // Degrees Celsius
320

321 /**
322 * Heated chamber watch settings (M141/M191).
323 */
324 #define WATCH_CHAMBER_TEMP_PERIOD 240 // Seconds
325 #define WATCH_CHAMBER_TEMP_INCREASE 1 // Degrees Celsius
326#endif

327

328 /**
329 * Thermal Protection parameters for the laser cooler.
330 */
331 #if ENABLED(THERMAL_PROTECTION_COOLER)
332 #define THERMAL_PROTECTION_COOLER_PERIOD 10 // Seconds
333 #define THERMAL_PROTECTION_COOLER_HYSTERESIS 3 // Degrees Celsius
334

335 /**
336 * Laser cooling watch settings (M143/M193).
337 */
338 #define WATCH_COOLER_TEMP_PERIOD 60 // Seconds
339 #define WATCH_COOLER_TEMP_INCREASE 3 // Degrees Celsius
340#endif

341

342 #if ENABLED(PIDTEMP)
343     // Add an experimental additional term to the heater power, proportional
344     // to the extrusion speed.
345     // A well-chosen Kc value should add just enough power to melt the
346     // increased material volume.
347     //#define PID_EXTRUSION_SCALING
348     #if ENABLED(PID_EXTRUSION_SCALING)
349         #define DEFAULT_Kc (100) // heating power = Kc * e_speed
350         #define LPQ_MAX_LEN 50
351     #endif

352     /**
353     * Add an experimental additional term to the heater power, proportional
354     // to the fan speed.
355     * A well-chosen Kf value should add just enough power to compensate for
356     // power-loss from the cooling fan.
357     * You can either just add a constant compensation with the DEFAULT_Kf
358     // value
359     * or follow the instruction below to get speed-dependent compensation.
360     *
361     * Constant compensation (use only with fanspeeds of 0% and 100%)
362     * -----
363     * A good starting point for the Kf-value comes from the calculation:
364     * kf = (power_fan * eff_fan) / power_heater * 255
365     * where eff_fan is between 0.0 and 1.0, based on fan-efficiency and
366     // airflow to the nozzle / heater.
367     *

```

```

363 * Example:
364 *   Heater: 40W, Fan: 0.1A * 24V = 2.4W, eff_fan = 0.8
365 *   Kf = (2.4W * 0.8) / 40W * 255 = 12.24
366 *
367 * Fan-speed dependent compensation
368 * -----
369 * 1. To find a good Kf value, set the hotend temperature, wait for it to
370 settle, and enable the fan (100%).
371 *   Make sure PID_FAN_SCALING_LIN_FACTOR is 0 and
372 PID_FAN_SCALING_ALTERNATIVE_DEFINITION is not enabled.
373 *   If you see the temperature drop repeat the test, increasing the Kf
374 value slowly, until the temperature
375 *   drop goes away. If the temperature overshoots after enabling the
376 fan, the Kf value is too big.
377 * 2. Note the Kf-value for fan-speed at 100%
378 * 3. Determine a good value for PID_FAN_SCALING_MIN_SPEED, which is
379 around the speed, where the fan starts moving.
380 * 4. Repeat step 1. and 2. for this fan speed.
381 * 5. Enable PID_FAN_SCALING_ALTERNATIVE_DEFINITION and enter the two
382 identified Kf-values in
383 *   PID_FAN_SCALING_AT_FULL_SPEED and PID_FAN_SCALING_AT_MIN_SPEED.
384 Enter the minimum speed in PID_FAN_SCALING_MIN_SPEED
385 */
386 // #define PID_FAN_SCALING
387 #if ENABLED(PID_FAN_SCALING)
388 // #define PID_FAN_SCALING_ALTERNATIVE_DEFINITION
389 // #if ENABLED(PID_FAN_SCALING_ALTERNATIVE_DEFINITION)
390 //   // The alternative definition is used for an easier configuration.
391 //   // Just figure out Kf at fullspeed (255) and
392 PID_FAN_SCALING_MIN_SPEED.
393 //     // DEFAULT_Kf and PID_FAN_SCALING_LIN_FACTOR are calculated
394 accordingly.
395
396 // #define PID_FAN_SCALING_AT_FULL_SPEED 13.0
397 // ==PID_FAN_SCALING_LIN_FACTOR*255+DEFAULT_Kf
398 // #define PID_FAN_SCALING_AT_MIN_SPEED 6.0
399 // ==PID_FAN_SCALING_LIN_FACTOR*PID_FAN_SCALING_MIN_SPEED+DEFAULT_Kf
400 // #define PID_FAN_SCALING_MIN_SPEED      10.0          // Minimum fan speed
401 at which to enable PID_FAN_SCALING
402
403 // #define DEFAULT_Kf (255.0*PID_FAN_SCALING_AT_MIN_SPEED-
404 PID_FAN_SCALING_AT_FULL_SPEED*PID_FAN_SCALING_MIN_SPEED)/(255.0-
405 PID_FAN_SCALING_MIN_SPEED)
406 // #define PID_FAN_SCALING_LIN_FACTOR (PID_FAN_SCALING_AT_FULL_SPEED-
407 DEFAULT_Kf)/255.0
408
409 // #else
410 //   #define PID_FAN_SCALING_LIN_FACTOR (0)           // Power loss due
411 to cooling = Kf * (fan_speed)
412 //   #define DEFAULT_Kf 10                         // A constant value
413 added to the PID-tuner
414 //   #define PID_FAN_SCALING_MIN_SPEED 10          // Minimum fan
415 speed at which to enable PID_FAN_SCALING
416 // #endif
417 // #endif
418 #endif
419
420 /**
421 * Automatic Temperature Mode

```

```

403     * AUTOMATIC TEMPERATURE MODE
404     *
405     * Dynamically adjust the hotend target temperature based on planned E
406     * moves.
407     *
408     * (Contrast with PID_EXTRUSION_SCALING, which tracks E movement and adjusts
409     * PID
410     * behavior using an additional kC value.)
411     *
412     * Autotemp is calculated by (mintemp + factor * mm_per_sec), capped to
413     * maxtemp.
414     *
415 #define AUTOTEMP
416 #if ENABLED(AUTOTEMP)
417     #define AUTOTEMP_OLDWEIGHT    0.98
418     // Turn on AUTOTEMP on M104/M109 by default using proportions set here
419     //#define AUTOTEMP_PROPORTIONAL
420     #if ENABLED(AUTOTEMP_PROPORTIONAL)
421         #define AUTOTEMP_MIN_P      0 // (°C) Added to the target temperature
422         #define AUTOTEMP_MAX_P      5 // (°C) Added to the target temperature
423         #define AUTOTEMP_FACTOR_P   1 // Apply this F parameter by default
424     (overridden by M104/M109 F)
425     #endif
426 #endif
427
428 // Show Temperature ADC value
429 // Enable for M105 to include ADC values read from temperature sensors.
430 // #define SHOW_TEMP_ADC_VALUES
431
432 /**
433     * High Temperature Thermistor Support
434     *
435     * Thermistors able to support high temperature tend to have a hard time
436     * getting
437     * good readings at room and lower temperatures. This means
438     * TEMP_SENSOR_X_RAW_LO_TEMP
439     * will probably be caught when the heating element first turns on during
440     * the
441     * preheating process, which will trigger a min_temp_error as a safety
442     * measure
443     * and force stop everything.
444     * To circumvent this limitation, we allow for a preheat time (during which,
445     * min_temp_error won't be triggered) and add a min_temp buffer to handle
446     * aberrant readings.
447     *
448     * If you want to enable this feature for your hotend thermistor(s)
449     * uncomment and set values > 0 in the constants below
450     */
451
452 // The number of consecutive low temperature errors that can occur
453 // before a min_temp_error is triggered. (Shouldn't be more than 10.)
454 // #define MAX_CONSECUTIVE_LOW_TEMPERATURE_ERROR_ALLOWED 0
455
456 // The number of milliseconds a hotend will preheat before starting to check
457 // the temperature. This value should NOT be set to the time it takes the

```

```

453 // hot end to reach the target temperature, but the time it takes to reach
454 // the minimum temperature your thermistor can read. The lower the
455 // better/safer.
456 // This shouldn't need to be more than 30 seconds (30000)
457 // #define MILLISECONDS_PREHEAT_TIME 0
458
459 // @section extruder
460
461 // Extruder runout prevention.
462 // If the machine is idle and the temperature over MINTEMP
463 // then extrude some filament every couple of SECONDS.
464 // #define EXTRUDER_RUNOUT_PREVENT
465 #if ENABLED(EXTRUDER_RUNOUT_PREVENT)
466     #define EXTRUDER_RUNOUT_MINTEMP 190
467     #define EXTRUDER_RUNOUT_SECONDS 30
468     #define EXTRUDER_RUNOUT_SPEED 1500 // (mm/min)
469     #define EXTRUDER_RUNOUT_EXTRUDE 5 // (mm)
470 #endif
471
472 /**
473 * Hotend Idle Timeout
474 * Prevent filament in the nozzle from charring and causing a critical jam.
475 */
476 // #define HOTEND_IDLE_TIMEOUT
477 #if ENABLED(HOTEND_IDLE_TIMEOUT)
478     #define HOTEND_IDLE_TIMEOUT_SEC (5*60)      // (seconds) Time without
479     // extruder movement to trigger protection
480     #define HOTEND_IDLE_MIN_TRIGGER 180        // (°C) Minimum temperature to
481     // enable hotend protection
482     #define HOTEND_IDLE_NOZZLE_TARGET 0         // (°C) Safe temperature for the
483     // nozzle after timeout
484     #define HOTEND_IDLE_BED_TARGET 0           // (°C) Safe temperature for the
485     // bed after timeout
486 #endif
487
488 // @section temperature
489
490 // Calibration for AD595 / AD8495 sensor to adjust temperature measurements.
491 // The final temperature is calculated as (measuredTemp * GAIN) + OFFSET.
492 #define TEMP_SENSOR_AD595_OFFSET 0.0
493 #define TEMP_SENSOR_AD595_GAIN 1.0
494 #define TEMP_SENSOR_AD8495_OFFSET 0.0
495 #define TEMP_SENSOR_AD8495_GAIN 1.0
496
497 /**
498 * Controller Fan
499 * To cool down the stepper drivers and MOSFETs.
500 *
501 * The fan turns on automatically whenever any driver is enabled and turns
502 * off (or reduces to idle speed) shortly after drivers are turned off.
503 */
504 // #define USE_CONTROLLER_FAN
505 #if ENABLED(USE_CONTROLLER_FAN)
506     // #define CONTROLLER_FAN_PIN -1           // Set a custom pin for the
507     // controller fan
508     // #define CONTROLLER_FAN_USE_Z_ONLY      // With this option only the Z
509     // axis is considered
510     // #define CONTROLLER_FAN_IGNORE_Z        // Ignore Z stepper. Useful when
511     // stepper timeout is disabled

```

```

504 // Stepper timeout is unused.
505 #define CONTROLLERFAN_SPEED_MIN          0 // (0-255) Minimum speed. (If
set below this value the fan is turned off.)
506 #define CONTROLLERFAN_SPEED_ACTIVE      255 // (0-255) Active speed, used
when any motor is enabled
507 #define CONTROLLERFAN_SPEED_IDLE        0 // (0-255) Idle speed, used when
motors are disabled
508 #define CONTROLLERFAN_IDLE_TIME        60 // (seconds) Extra time to keep
the fan running after disabling motors
509 // Use TEMP_SENSOR_BOARD as a trigger for enabling the controller fan
510 // #define CONTROLLER_FAN_MIN_BOARD_TEMP 40 // (°C) Turn on the fan if the
board reaches this temperature
511
512 // #define CONTROLLER_FAN_EDITABLE           // Enable M710 configurable
settings
513 #if ENABLED(CONTROLLER_FAN_EDITABLE)
514   #define CONTROLLER_FAN_MENU            // Enable the Controller Fan
submenu
515   #endif
516 #endif
517
518 // When first starting the main fan, run it at full speed for the
519 // given number of milliseconds. This gets the fan spinning reliably
520 // before setting a PWM value. (Does not work with software PWM for fan on
Sanguinololu)
521 //#define FAN_KICKSTART_TIME 100
522
523 // Some coolers may require a non-zero "off" state.
524 //#define FAN_OFF_PWM 1
525
526 /**
527 * PWM Fan Scaling
528 *
529 * Define the min/max speeds for PWM fans (as set with M106).
530 *
531 * With these options the M106 0-255 value range is scaled to a subset
532 * to ensure that the fan has enough power to spin, or to run lower
533 * current fans with higher current. (e.g., 5V/12V fans with 12V/24V)
534 * Value 0 always turns off the fan.
535 *
536 * Define one or both of these to override the default 0-255 range.
537 */
538 //#define FAN_MIN_PWM 50
539 //#define FAN_MAX_PWM 128
540
541 /**
542 * FAST PWM FAN Settings
543 *
544 * Use to change the FAST FAN PWM frequency (if enabled in Configuration.h)
545 * Combinations of PWM Modes, prescale values and TOP resolutions are used
internally to produce a
546 * frequency as close as possible to the desired frequency.
547 *
548 * FAST_PWM_FAN_FREQUENCY [undefined by default]
549 * Set this to your desired frequency.
550 * If left undefined this defaults to F = F_CPU/(2*255*1)
551 * i.e., F = 31.4kHz on 16MHz microcontrollers or F = 39.2kHz on 20MHz
microcontrollers.

```

```

552 * These defaults are the same as with the old FAST_PWM_FAN implementation
553 - no migration is required
554 * NOTE: Setting very low frequencies (< 10 Hz) may result in unexpected
555 timer behavior.
556 *
557 * USE_OCR2A_AS_TOP [undefined by default]
558 * Boards that use TIMER2 for PWM have limitations resulting in only a few
559 possible frequencies on TIMER2:
560 * 16MHz MCUs: [62.5KHz, 31.4KHz (default), 7.8KHz, 3.92KHz, 1.95KHz,
561 977Hz, 488Hz, 244Hz, 60Hz, 122Hz, 30Hz]
562 * 20MHz MCUs: [78.1KHz, 39.2KHz (default), 9.77KHz, 4.9KHz, 2.44KHz,
563 1.22KHz, 610Hz, 305Hz, 153Hz, 76Hz, 38Hz]
564 * A greater range can be achieved by enabling USE_OCR2A_AS_TOP. But note
565 that this option blocks the use of
566 * PWM on pin OC2A. Only use this option if you don't need PWM on OC2A.
567 (Check your schematic.)
568 * USE_OCR2A_AS_TOP sacrifices duty cycle control resolution to achieve
569 this broader range of frequencies.
570 */
571 #if ENABLED(FAST_PWM_FAN)
572     // #define FAST_PWM_FAN_FREQUENCY 31400
573     // #define USE_OCR2A_AS_TOP
574 #endif
575 /**
576 * Use one of the PWM fans as a redundant part-cooling fan
577 */
578 // #define REDUNDANT_PART_COOLING_FAN 2 // Index of the fan to sync with FAN
579 0.
580
581 // @section extruder
582
583 /**
584 * Extruder cooling fans
585 *
586 * Extruder auto fans automatically turn on when their extruders'
587 * temperatures go above EXTRUDER_AUTO_FAN_TEMPERATURE.
588 *
589 * Your board's pins file specifies the recommended pins. Override those
590 here
591 * or set to -1 to disable completely.
592 *
593 * Multiple extruders can be assigned to the same pin in which case
594 * the fan will turn on when any selected extruder is above the threshold.
595 */
596 #define E0_AUTO_FAN_PIN -1
597 #define E1_AUTO_FAN_PIN -1
598 #define E2_AUTO_FAN_PIN -1
599 #define E3_AUTO_FAN_PIN -1
600 #define E4_AUTO_FAN_PIN -1
601 #define E5_AUTO_FAN_PIN -1
602 #define E6_AUTO_FAN_PIN -1
603 #define E7_AUTO_FAN_PIN -1
604 #define CHAMBER_AUTO_FAN_PIN -1
605 #define COOLER_AUTO_FAN_PIN -1
606 #define COOLER_FAN_PIN -1
607
608 /**
609 * EXTRUDER_AUTO_FAN_TEMPERATURE 50
610 * #define EXTRUDER_AUTO_FAN_CDEF 0EE // 0EE == full speed

```

```

599 #define EXTRUDER_AUTO_FAN_SPEED 255 // 255 == full speed
600 #define CHAMBER_AUTO_FAN_TEMPERATURE 30
601 #define CHAMBER_AUTO_FAN_SPEED 255
602 #define COOLER_AUTO_FAN_TEMPERATURE 18
603 #define COOLER_AUTO_FAN_SPEED 255
604
605 /**
606 * Part-Cooling Fan Multiplexer
607 *
608 * This feature allows you to digitally multiplex the fan output.
609 * The multiplexer is automatically switched at tool-change.
610 * Set FANMUX[012]_PINs below for up to 2, 4, or 8 multiplexed fans.
611 */
612
613 #define FANMUX0_PIN -1
614 #define FANMUX1_PIN -1
615 #define FANMUX2_PIN -1
616
617 /**
618 * M355 Case Light on-off / brightness
619 */
620 // #define CASE_LIGHT_ENABLE
621 #if ENABLED(CASE_LIGHT_ENABLE)
622     // #define CASE_LIGHT_PIN 4           // Override the default pin if
623     // needed
624     #define INVERT_CASE_LIGHT false    // Set true if Case Light is
625     ON when pin is LOW
626     #define CASE_LIGHT_DEFAULT_ON true // Set default power-up state
627     ON
628     #define CASE_LIGHT_DEFAULT_BRIGHTNESS 105 // Set default power-up
629     brightness (0-255, requires PWM pin)
630     // #define CASE_LIGHT_NO_BRIGHTNESS // Disable brightness control.
631     // Enable for non-PWM lighting.
632     // #define CASE_LIGHT_MAX_PWM 128   // Limit PWM duty cycle (0-
633     // 255)
634     // #define CASE_LIGHT_MENU        // Add Case Light options to
635     // the LCD menu
636     #if ENABLED(NEOPIXEL_LED)
637         // #define CASE_LIGHT_USE_NEOPIXEL // Use NeoPixel LED as case
638         light
639         #endif
640         #if EITHER(RGB_LED, RGBW_LED)
641             // #define CASE_LIGHT_USE_RGB_LED // Use RGB / RGBW LED as case
642             light
643             #endif
644             #if EITHER(CASE_LIGHT_USE_NEOPIXEL, CASE_LIGHT_USE_RGB_LED)
645                 #define CASE_LIGHT_DEFAULT_COLOR { 255, 255, 255, 255 } // { Red, Green,
646                 Blue, White }
647                 #endif
648             #endif
649
650 // @section homing
651
652 // If you want endstops to stay on (by default) even when not homing
653 // enable this option. Override at any time with M120, M121.
654 // #define ENDSTOPS_ALWAYS_ON_DEFAULT
655
656 // @section extras
657
658 // #define Z_LATE_ENABLE // Enable Z the last moment. Needed if your Z driver

```

```

overheats.

649
650 // Employ an external closed loop controller. Override pins here if needed.
651 // #define EXTERNAL_CLOSED_LOOP_CONTROLLER
652 #if ENABLED(EXTERNAL_CLOSED_LOOP_CONTROLLER)
653     // #define CLOSED_LOOP_ENABLE_PIN -1
654     // #define CLOSED_LOOP_MOVE_COMPLETE_PIN -1
655 #endif
656
657 /**
658 * Dual Steppers / Dual Endstops
659 *
660 * This section will allow you to use extra E drivers to drive a second
661 motor for X, Y, or Z axes.
662 *
663 * For example, set X_DUAL_STEPPER_DRIVERS setting to use a second motor. If
664 the motors need to
665 * spin in opposite directions set INVERT_X2_VS_X_DIR. If the second motor
666 needs its own endstop
667 * set X_DUAL_ENDSTOPS. This can adjust for "racking." Use X2_USE_ENDSTOP to
668 set the endstop plug
669 * that should be used for the second endstop. Extra endstops will appear in
670 the output of 'M119'.
671 *
672 * Use X_DUAL_ENDSTOP_ADJUSTMENT to adjust for mechanical imperfection.
673 After homing both motors
674 * this offset is applied to the X2 motor. To find the offset home the X
675 axis, and measure the error
676 * in X2. Dual endstop offsets can be set at runtime with 'M666 X<offset>
677 Y<offset> Z<offset>'.
678 */
679
680 // #define X_DUAL_STEPPER_DRIVERS
681 #if ENABLED(X_DUAL_STEPPER_DRIVERS)
682     // #define INVERT_X2_VS_X_DIR // Enable if X2 direction signal is
683     // opposite to X
684     // #define X_DUAL_ENDSTOPS
685     #if ENABLED(X_DUAL_ENDSTOPS)
686         #define X2_USE_ENDSTOP_XMAX_
687         #define X2_ENDSTOP_ADJUSTMENT 0
688     #endif
689 #endif
690
691 // #define Y_DUAL_STEPPER_DRIVERS
692 #if ENABLED(Y_DUAL_STEPPER_DRIVERS)
693     // #define INVERT_Y2_VS_Y_DIR // Enable if Y2 direction signal is
694     // opposite to Y
695     // #define Y_DUAL_ENDSTOPS
696     #if ENABLED(Y_DUAL_ENDSTOPS)
697         #define Y2_USE_ENDSTOP_YMAX_
698         #define Y2_ENDSTOP_ADJUSTMENT 0
699     #endif
700 #endif
701
702 /**
703 // For Z set the number of stepper drivers
704 //
705 #define NUM_Z_STEPPER_DRIVERS 1 // (1-4) Z options change based on how
706

```

```
 711 //#
 712 #endif
 713
 714 // Drive the E axis with two synchronized steppers
 715 //#define E_DUAL_STEPPER_DRIVERS
 716 #if ENABLED(E_DUAL_STEPPER_DRIVERS)
 717   //#define INVERT_E1_VS_E0_DIR // Enable if the E motors need opposite
 718   DIR states
 719 #endif
 720
 721 /**
 722  * Dual X Carriage
 723  *
 724  * This setup has two X carriages that can move independently, each with its
 725  * own hotend.
 726  * The carriages can be used to print an object with two colors or
 727  * materials, or in
 728  * "duplication mode" it can print two identical or X-mirrored objects
 729  * simultaneously.
 730  * The inactive carriage is parked automatically to prevent oozing.
 731  * X1 is the left carriage, X2 the right. They park and home at opposite
 732  * ends of the X axis.
 733  * By default the X2 stepper is assigned to the first unused E plug on the
 734  * board.
 735  *
 736  * The following Dual X Carriage modes can be selected with M605 S<mode>:
 737  *
 738  * 0 : (FULL_CONTROL) The slicer has full control over both X-carriages
 739  * and can achieve optimal travel
 740  *      results as long as it supports dual X-carriages. (M605 S0)
 741  *
 742  * 1 : (AUTO_PARK) The firmware automatically parks and unparks the X-
 743  * carriages on tool-change so
 744  *      that additional slicer support is not required. (M605 S1)
 745  *
 746  * 2 : (DUPLICATION) The firmware moves the second X-carriage and extruder
 747  * in synchronization with
 748  *      the first X-carriage and extruder, to print 2 copies of the same
 749  * object at the same time.
```

```

744 *      Set the constant X-offset and temperature differential with M605 S2
745 X[offs] R[deg] and
746 *      follow with M605 S2 to initiate duplicated movement.
747 *
748 * 3 : (MIRRORED) Formbot/Vivedino-inspired mirrored mode in which the
749 second extruder duplicates
750 *      the movement of the first except the second extruder is reversed in
751 the X axis.
752 *      Set the initial X offset and temperature differential with M605 S2
753 X[offs] R[deg] and
754 *      follow with M605 S3 to initiate mirrored movement.
755 */
756 //#define DUAL_X_CARRIAGE
757 #if ENABLED(DUAL_X_CARRIAGE)
758     #define X1_MIN_POS X_MIN_POS // Set to X_MIN_POS
759     #define X1_MAX_POS X_BED_SIZE // Set a maximum so the first X-carriage
760 can't hit the parked second X-carriage
761     #define X2_MIN_POS 80 // Set a minimum to ensure the second X-
762 carriage can't hit the parked first X-carriage
763     #define X2_MAX_POS 353 // Set this to the distance between
764 toolheads when both heads are homed
765     #define X2_HOME_DIR 1 // Set to 1. The second X-carriage always
766 homes to the maximum endstop position
767     #define X2_HOME_POS X2_MAX_POS // Default X2 home position. Set to
768 X2_MAX_POS.
769                         // However: In this mode the HOTEND_OFFSET_X value for
770 the second extruder provides a software
771                         // override for X2_HOME_POS. This also allow
772 recalibration of the distance between the two endstops
773                         // without modifying the firmware (through the "M218
774 T1 X???" command).
775                         // Remember: you should set the second extruder x-
776 offset to 0 in your slicer.
777
778 // This is the default power-up mode which can be later using M605.
779 #define DEFAULT_DUAL_X_CARRIAGE_MODE DXC_AUTO_PARK_MODE
780
781 // Default x offset in duplication mode (typically set to half print bed
782 width)
783 #define DEFAULT_DUPLICATION_X_OFFSET 100
784
785 // Default action to execute following M605 mode change commands.
786 Typically G28X to apply new mode.
787     //#define EVENT_GCODE_IDEX_AFTER_MODECHANGE "G28X"
788 #endif
789
790 // Activate a solenoid on the active extruder with M380. Disable all with
791 M381.
792 // Define SOL0_PIN, SOL1_PIN, etc., for each extruder that has a solenoid.
793 // #define EXT_SOLENOID
794
795 // @section homing
796
797 /**
798 * Homing Procedure
799 * Homing (G28) does an indefinite move towards the endstops to establish
800 * the position of the toolhead relative to the workspace.
801 */

```

```
/88
787 // #define SENSORLESS_BACKOFF_MM { 2, 2, 0 } // (mm) Backoff from endstops
before sensorless homing
788
789 #define HOMING_BUMP_MM      { 5, 5, 2 }          // (mm) Backoff from endstops
after first bump
790 #define HOMING_BUMP_DIVISOR { 2, 2, 4 }          // Re-Bump Speed Divisor
(Divides the Homing Feedrate)
791
792 // #define HOMING_BACKOFF_POST_MM { 2, 2, 2 } // (mm) Backoff from endstops
after homing
793
794 // #define QUICK_HOME           // If G28 contains XY do a
diagonal move first
795 // #define HOME_Y_BEFORE_X     // If G28 contains XY home Y
before X
796 // #define HOME_Z_FIRST        // Home Z first. Requires a Z-
MIN endstop (not a probe).
797 // #define CODEPENDENT_XY_HOMING // If X/Y can't home without
homing Y/X first
798
799 // @section bltouch
800
801 #if ENABLED(BLTOUCH)
802 /**
803  * Either: Use the defaults (recommended) or: For special purposes, use
the following DEFINES
804  * Do not activate settings that the probe might not understand. Clones
might misunderstand
805  * advanced commands.
806  *
807  * Note: If the probe is not deploying, do a "Reset" and "Self-Test" and
then check the
808  * wiring of the BROWN, RED and ORANGE wires.
809  *
810  * Note: If the trigger signal of your probe is not being recognized, it
has been very often
811  * because the BLACK and WHITE wires needed to be swapped. They are
not "interchangeable"
812  * like they would be with a real switch. So please check the wiring
first.
813  *
814  * Settings for all BLTouch and clone probes:
815 */
816
817 // Safety: The probe needs time to recognize the command.
818 // Minimum command delay (ms). Enable and increase if needed.
819 // #define BLTOUCH_DELAY 500
820
821 /**
822  * Settings for BLTOUCH Classic 1.2, 1.3 or BLTouch Smart 1.0, 2.0, 2.2,
3.0, 3.1, and most clones:
823 */
824
825 // Feature: Switch into SW mode after a deploy. It makes the output pulse
longer. Can be useful
826 // in special cases, like noisy or filtered input configurations.
827 // #define BLTOUCH_FORCE_SW_MODE
828
```

```

829 /**
830  * Settings for BLTouch Smart 3.0 and 3.1
831  * Summary:
832  *   - Voltage modes: 5V and OD (open drain - "logic voltage free") output
833  * modes
834  *   - High-Speed mode
835  *   - Disable LCD voltage options
836  */
837 /**
838  * Danger: Don't activate 5V mode unless attached to a 5V-tolerant
839  * controller!
840  * V3.0 or 3.1: Set default mode to 5V mode at Marlin startup.
841  * If disabled, OD mode is the hard-coded default on 3.0
842  * On startup, Marlin will compare its eeprom to this value. If the
843  * selected mode
844  * differs, a mode set eeprom write will be completed at initialization.
845  * Use the option below to force an eeprom write to a V3.1 probe
846  * regardless.
847 */
848 //">#define BLTOUCH_SET_5V_MODE
849
850 /**
851  * Safety: Activate if connecting a probe with an unknown voltage mode.
852  * V3.0: Set a probe into mode selected above at Marlin startup. Required
853  * for 5V mode on 3.0
854  * V3.1: Force a probe with unknown mode into selected mode at Marlin
855  * startup ( = Probe EEPROM write )
856  * To preserve the life of the probe, use this once then turn it off and
857  * re-flash.
858 */
859 //">#define BLTOUCH_FORCE_MODE_SET
860
861 /**
862  * Use "HIGH SPEED" mode for probing.
863  * Danger: Disable if your probe sometimes fails. Only suitable for stable
864  * well-adjusted systems.
865  * This feature was designed for Deltabots with very fast Z moves;
866  * however, higher speed Cartesians
867  * might be able to use it. If the machine can't raise Z fast enough the
868  * BLTouch may go into ALARM.
869 */
870 //">#define BLTOUCH_HS_MODE
871
872 // Safety: Enable voltage mode settings in the LCD menu.
873 //">#define BLTOUCH_LCD_VOLTAGE_MENU
874
875 #endif // BLTOUCH
876
877 // @section extras
878
879 /**
880  * Z Steppers Auto-Alignment
881  * Add the G34 command to align multiple Z steppers using a bed probe.
882  */
883 //">#define Z_STEPPER_AUTO_ALIGN
884 #if ENABLED(Z_STEPPER_AUTO_ALIGN)
885  // Define probe X and Y positions for Z1, Z2 [, Z3 [, Z4]]

```

```

877 // If not defined, probe limits will be used.
878 // Override with 'M422 S<index> X<pos> Y<pos>'  

879 // #define Z_STEPPER_ALIGN_XY { { 10, 190 }, { 100, 10 }, { 190, 190 } }  

880  

881 /**
882 * Orientation for the automatically-calculated probe positions.
883 * Override Z stepper align points with 'M422 S<index> X<pos> Y<pos>'  

884 *
885 * 2 Steppers: (0) (1)
886 * | 1 2 | 2 |
887 * | | 1 | |
888 *
889 *
890 * 3 Steppers: (0) (1) (2) (3)
891 * | 3 | 1 | 2 | 1 | 3 | 2 |
892 * | | 2 | 2 | 3 | 3 | 1 | 2 |
893 * | 1 2 | 2 | 3 | 3 | 4 | 1 | 2 |
894 *
895 * 4 Steppers: (0) (1) (2) (3)
896 * | 4 | 3 | 1 | 4 | 2 | 1 | 3 | 2 |
897 * | | 2 | 2 | 3 | 3 | 4 | 4 | 1 | 2 |
898 * |
899 */
900 #ifndef Z_STEPPER_ALIGN_XY
901     // #define Z_STEPPERS_ORIENTATION 0
902 #endif
903
904 // Provide Z stepper positions for more rapid convergence in bed
905 alignment.
906 // Requires triple stepper drivers (i.e., set NUM_Z_STEPPER_DRIVERS to 3)
907 // #define Z_STEPPER_ALIGN_KNOWN_STEPPER_POSITIONS
908 #if ENABLED(Z_STEPPER_ALIGN_KNOWN_STEPPER_POSITIONS)
909     // Define Stepper XY positions for Z1, Z2, Z3 corresponding to
910     // the Z screw positions in the bed carriage.
911     // Define one position per Z stepper in stepper driver order.
912     #define Z_STEPPER_ALIGN_STEPPER_XY { { 210.7, 102.5 }, { 152.6, 220.0 },
913     { 94.5, 102.5 } }
914 #else
915     // Amplification factor. Used to scale the correction step up or down in
916     // case
917     // the stepper (spindle) position is farther out than the test point.
918     #define Z_STEPPER_ALIGN_AMP 1.0          // Use a value > 1.0 NOTE: This
919     // may cause instability!
920 #endif
921
922     // On a 300mm bed a 5% grade would give a misalignment of ~1.5cm
923     #define G34_MAX_GRADE      5          // (%) Maximum incline that G34
924     // will handle
925     #define Z_STEPPER_ALIGN_ITERATIONS 5    // Number of iterations to apply
926     // during alignment
927     #define Z_STEPPER_ALIGN_ACC        0.02 // Stop iterating early if the
928     // accuracy is better than this
929     #define RESTORE_LEVELING_AFTER_G34 // Restore leveling after G34 is
930     // done?
931     // After G34, re-home Z (G28 Z) or just calculate it from the last probe
932     // heights?
933     // Re-homing might be more precise in reproducing the actual 'G28 Z'
934     // homing height, especially on an uneven bed.
935     #define HOME_AFTER_G34

```

```
926 #endif
927 //
928 // Add the G35 command to read bed corners to help adjust screws. Requires a
929 // bed probe.
930 //
931 //#define ASSISTED_TRAMMING
932 #if ENABLED(ASSISTED_TRAMMING)
933
934     // Define positions for probe points.
935     #define TRAMMING_POINT_XY { { 20, 20 }, { 180, 20 }, { 180, 180 }, { 20,
936     180 } }
937
938     // Define position names for probe points.
939     #define TRAMMING_POINT_NAME_1 "Front-Left"
940     #define TRAMMING_POINT_NAME_2 "Front-Right"
941     #define TRAMMING_POINT_NAME_3 "Back-Right"
942     #define TRAMMING_POINT_NAME_4 "Back-Left"
943
944     #define RESTORE_LEVELING_AFTER_G35      // Enable to restore leveling setup
945     after operation
946     //##define REPORT_TRAMMING_MM          // Report Z deviation (mm) for each
947     point relative to the first
948
949     //##define ASSISTED_TRAMMING_WIZARD    // Add a Tramming Wizard to the LCD
950     menu
951
952     //##define ASSISTED_TRAMMING_WAIT_POSITION { X_CENTER, Y_CENTER, 30 } //
953     Move the nozzle out of the way for adjustment
954
955 /**
956 * Screw thread:
957 *   M3: 30 = Clockwise, 31 = Counter-Clockwise
958 *   M4: 40 = Clockwise, 41 = Counter-Clockwise
959 *   M5: 50 = Clockwise, 51 = Counter-Clockwise
960 */
961 #define TRAMMING_SCREW_THREAD 30
962
963 #endif
964
965 // @section motion
966
967 #define AXIS_RELATIVE_MODES { false, false, false, false }
968
969 // Add a Duplicate option for well-separated conjoined nozzles
970 //##define MULTI_NOZZLE_DUPLICATION
971
972 // By default pololu step drivers require an active high signal. However,
973 // some high power drivers require an active low signal as step.
974 #define INVERT_X_STEP_PIN false
975 #define INVERT_Y_STEP_PIN false
976 #define INVERT_Z_STEP_PIN false
977 #define INVERT_I_STEP_PIN false
978 #define INVERT_J_STEP_PIN false
979 #define INVERT_K_STEP_PIN false
980 #define INVERT_E_STEP_PIN false
981
982 /**
983 ...
```

```

977 * Idle Stepper Shutdown
978 * Set DISABLE_INACTIVE_? 'true' to shut down axis steppers after an idle
979 period.
980 * The Deactive Time can be overridden with M18 and M84. Set to 0 for No
981 Timeout.
982 */
983 #define DEFAULT_STEPPER_DEACTIVE_TIME 120
984 #define DISABLE_INACTIVE_X true
985 #define DISABLE_INACTIVE_Y true
986 #define DISABLE_INACTIVE_Z true // Set 'false' if the nozzle could fall
987 onto your printed part!
988 #define DISABLE_INACTIVE_I true
989 #define DISABLE_INACTIVE_J true
990 #define DISABLE_INACTIVE_K true
991 #define DISABLE_INACTIVE_E true
992
993 // Default Minimum Feedrates for printing and travel moves
994 #define DEFAULT_MINIMUMFEEDRATE      0.0      // (mm/s) Minimum feedrate.
995 Set with M205 S.
996 #define DEFAULT_MINTRAVELFEEDRATE    0.0      // (mm/s) Minimum travel
997 feedrate. Set with M205 T.
998
999 // Minimum time that a segment needs to take as the buffer gets emptied
1000 #define DEFAULT_MINSEGMENTTIME     20000    // (μs) Set with M205 B.
1001
1002 // Slow down the machine if the lookahead buffer is (by default) half full.
1003 // Increase the slowdown divisor for larger buffer sizes.
1004 #define SLOWDOWN
1005 #if ENABLED(SLOWDOWN)
1006   #define SLOWDOWN_DIVISOR 2
1007 #endif
1008
1009 /**
1010  * XY Frequency limit
1011  * Reduce resonance by limiting the frequency of small zigzag infill moves.
1012  * See https://hydraraptor.blogspot.com/2010/12/frequency-limit.html
1013  * Use M201 F<freq> G<min%> to change limits at runtime.
1014  */
1015 // #define XY_FREQUENCY_LIMIT      10 // (Hz) Maximum frequency of small
1016 // zigzag infill moves. Set with M201 F<hertz>.
1017 #ifdef XY_FREQUENCY_LIMIT
1018   #define XY_FREQUENCY_MIN_PERCENT 5 // (percent) Minimum FR percentage to
1019   // apply. Set with M201 G<min%>.
1020 #endif
1021
1022 // Minimum planner junction speed. Sets the default minimum speed the
1023 // planner plans for at the end
1024 // of the buffer and all stops. This should not be much greater than zero
1025 // and should only be changed
1026 // if unwanted behavior is observed on a user's machine when running at very
1027 // slow speeds.
1028 #define MINIMUM_PLANNER_SPEED 0.05 // (mm/s)
1029
1030 //
1031 // Backlash Compensation
1032 // Adds extra movement to axes on direction-changes to account for backlash.
1033 //
1034 // #define BACKLASH_COMPENSATION
1035 #if FNAR(FBACKLASH_COMPENSATION)

```

```

1026 // Define values for backlash distance and correction.
1027 // If BACKLASH_GCODE is enabled these values are the defaults.
1028 #define BACKLASH_DISTANCE_MM { 0, 0, 0 } // (mm) One value for each linear
axis
1029 #define BACKLASH_CORRECTION 0.0 // 0.0 = no correction; 1.0 =
full correction
1030
1031 // Add steps for motor direction changes on CORE kinematics
1032 //#define CORE_BACKLASH
1033
1034 // Set BACKLASH_SMOOTHING_MM to spread backlash correction over multiple
segments
1035 // to reduce print artifacts. (Enabling this is costly in memory and
computation!)
1036 //#define BACKLASH_SMOOTHING_MM 3 // (mm)
1037
1038 // Add runtime configuration and tuning of backlash values (M425)
1039 //#define BACKLASH_GCODE
1040
1041 #if ENABLED(BACKLASH_GCODE)
1042     // Measure the Z backlash when probing (G29) and set with "M425 Z"
1043     #define MEASURE_BACKLASH_WHEN_PROBING
1044
1045     #if ENABLED(MEASURE_BACKLASH_WHEN_PROBING)
1046         // When measuring, the probe will move up to
BACKLASH_MEASUREMENT_LIMIT
1047         // mm away from point of contact in BACKLASH_MEASUREMENT_RESOLUTION
1048         // increments while checking for the contact to be broken.
1049         #define BACKLASH_MEASUREMENT_LIMIT 0.5 // (mm)
1050         #define BACKLASH_MEASUREMENT_RESOLUTION 0.005 // (mm)
1051         #define BACKLASH_MEASUREMENT_FEEDRATE Z_PROBE_FEEDRATE_SLOW // /
(mm/min)
1052     #endif
1053     #endif
1054 #endif
1055
1056 /**
1057 * Automatic backlash, position and hotend offset calibration
1058 *
1059 * Enable G425 to run automatic calibration using an electrically-
1060 * conductive cube, bolt, or washer mounted on the bed.
1061 *
1062 * G425 uses the probe to touch the top and sides of the calibration object
1063 * on the bed and measures and/or correct positional offsets, axis backlash
1064 * and hotend offsets.
1065 *
1066 * Note: HOTEND_OFFSET and CALIBRATION_OBJECT_CENTER must be set to within
1067 * ±5mm of true values for G425 to succeed.
1068 */
1069 //#define CALIBRATION_GCODE
1070 #if ENABLED(CALIBRATION_GCODE)
1071
1072     //#define CALIBRATION_SCRIPT_PRE "M117 Starting Auto-
Calibration\nT0\nG28\nG12\nM117 Calibrating..."
1073     //#define CALIBRATION_SCRIPT_POST "M500\nM117 Calibration data saved"
1074
1075     #define CALIBRATION_MEASUREMENT_RESOLUTION 0.01 // mm
1076

```

```

1077 #define CALIBRATION_FEEDRATE_SLOW           60      // mm/min
1078 #define CALIBRATION_FEEDRATE_FAST          1200    // mm/min
1079 #define CALIBRATION_FEEDRATE_TRAVEL        3000    // mm/min
1080
1081 // The following parameters refer to the conical section of the nozzle
1082 // tip.
1083 #define CALIBRATION_NOZZLE_TIP_HEIGHT       1.0     // mm
1084 #define CALIBRATION_NOZZLE_OUTER_DIAMETER    2.0     // mm
1085
1086 // Uncomment to enable reporting (required for "G425 V", but consumes
1087 // PROGMEM).
1088 // #define CALIBRATION_REPORTING
1089
1090 // The true location and dimension the cube/bolt/washer on the bed.
1091 #define CALIBRATION_OBJECT_CENTER   { 264.0, -22.0, -2.0 } // mm
1092 #define CALIBRATION_OBJECT_DIMENSIONS { 10.0, 10.0, 10.0 } // mm
1093
1094 // Comment out any sides which are unreachable by the probe. For best
1095 // auto-calibration results, all sides must be reachable.
1096 #define CALIBRATION_MEASURE_RIGHT
1097 #define CALIBRATION_MEASURE_FRONT
1098 #define CALIBRATION_MEASURE_LEFT
1099 #define CALIBRATION_MEASURE_BACK
1100
1101 // #define CALIBRATION_MEASURE_IMIN
1102 // #define CALIBRATION_MEASURE_IMAX
1103 // #define CALIBRATION_MEASURE_JMIN
1104 // #define CALIBRATION_MEASURE_JMAX
1105 // #define CALIBRATION_MEASURE_KMIN
1106 // #define CALIBRATION_MEASURE_KMAX
1107
1108 // Probing at the exact top center only works if the center is flat. If
1109 // probing on a screwhead or hollow washer, probe near the edges.
1110 // #define CALIBRATION_MEASURE_AT_TOP_EDGES
1111
1112 // Define the pin to read during calibration
1113 #ifndef CALIBRATION_PIN
1114     // #define CALIBRATION_PIN -1           // Define here to override the
1115     // default pin
1116     #define CALIBRATION_PIN_INVERTING false // Set to true to invert the
1117     // custom pin
1118     // #define CALIBRATION_PIN_PULLDOWN
1119     #define CALIBRATION_PIN_PULLUP
1120     #endif
1121 #endif
1122
1123 /**
1124 * Adaptive Step Smoothing increases the resolution of multi-axis moves,
1125 * particularly at step frequencies
1126 * below 1kHz (for AVR) or 10kHz (for ARM), where aliasing between axes in
1127 * multi-axis moves causes audible
1128 * vibration and surface artifacts. The algorithm adapts to provide the best
1129 * possible step smoothing at the
1130 * lowest stepping frequencies.
1131 */
1132 // #define ADAPTIVE_STEP_SMOOTHING
1133
1134 /**
1135 * Custom Microstepping

```

```

1129 * Override as-needed for your setup. Up to 3 MS pins are supported.
1130 */
1131 //#define MICROSTEP1 LOW,LOW,LOW
1132 //#define MICROSTEP2 HIGH,LOW,LOW
1133 //#define MICROSTEP4 LOW,HIGH,LOW
1134 //#define MICROSTEP8 HIGH,HIGH,LOW
1135 //#define MICROSTEP16 LOW,LOW,HIGH
1136 //#define MICROSTEP32 HIGH,LOW,HIGH
1137
1138 // Microstep settings (Requires a board with pins named X_MS1, X_MS2, etc.)
1139 #define MICROSTEP_MODES { 16, 16, 16, 16, 16, 16 } // [1,2,4,8,16]
1140
1141 /**
1142 * @section stepper motor current
1143 *
1144 * Some boards have a means of setting the stepper motor current via
1145 * firmware.
1146 *
1147 * The power on motor currents are set by:
1148 *   PWM_MOTOR_CURRENT - used by MINIRAMBO & ULTIMAIN_2
1149 *                           known compatible chips: A4982
1150 *   DIGIPOT_MOTOR_CURRENT - used by BQ_ZUM_MEGA_3D, RAMBO & SCOOVO_X9H
1151 *                           known compatible chips: AD5206
1152 *   DAC_MOTOR_CURRENT_DEFAULT - used by PRINTRBOARD_REVF & RIGIDBOARD_V2
1153 *                           known compatible chips: MCP4728
1154 *   DIGIPOT_I2C_MOTOR_CURRENTS - used by 5DPRINT, AZTEEG_X3_PRO,
1155 * AZTEEG_X5_MINI_WIFI, MIGHTYBOARD_REV
1156 *                           known compatible chips: MCP4451, MCP4018
1157 *
1158 * Motor currents can also be set by M907 - M910 and by the LCD.
1159 *   M907 - applies to all.
1160 *   M908 - BQ_ZUM_MEGA_3D, RAMBO, PRINTRBOARD_REVF, RIGIDBOARD_V2 &
1161 * SCOOVO_X9H
1162 *   M909, M910 & LCD - only PRINTRBOARD_REVF & RIGIDBOARD_V2
1163 */
1164
1165 /**
1166 * I2C-based DIGIPOTs (e.g., Azteeg X3 Pro)
1167 */
1168 //#define DIGIPOT_MCP4018           // Requires https://github.com/felias-
1169 fogg/SlowSoftI2CMaster
1170 //#define DIGIPOT_MCP4451
1171 #if EITHER(DIGIPOT_MCP4018, DIGIPOT_MCP4451)
1172     #define DIGIPOT_I2C_NUM_CHANNELS 8 // 5DPRINT:4    AZTEEG_X3_PRO:8
1173     MKS_SBASE:5    MIGHTYBOARD_REV:5
1174
1175     // Actual motor currents in Amps. The number of entries must match
1176     // DIGIPOT_I2C_NUM_CHANNELS.
1177     // These correspond to the physical drivers, so be mindful if the order is
1178     // changed.
1179     #define DIGIPOT_I2C_MOTOR_CURRENTS { 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,
1180     1.0 } // AZTEEG_X3_PRO

```

```

1176
1177 // #define DIGIPOT_USE_RAW_VALUES      // Use DIGIPOT_MOTOR_CURRENT raw wiper
   values (instead of A4988 motor currents)
1178
1179 /**
1180 * Common slave addresses:
1181 *
1182 *          A  (A shifted)    B  (B shifted)    IC
1183 * Smoothie           0x2C (0x58)        0x2D (0x5A)    MCP4451
1184 * AZTEEG_X3_PRO     0x2C (0x58)        0x2E (0x5C)    MCP4451
1185 * AZTEEG_X5_MINI    0x2C (0x58)        0x2E (0x5C)    MCP4451
1186 * AZTEEG_X5_MINI_WIFI 0x58            0x5C          MCP4451
1187 * MIGHTYBOARD_REVE  0x2F (0x5E)        0x5C          MCP4018
1188 */
1189 // #define DIGIPOT_I2C_ADDRESS_A 0x2C // Unshifted slave address for first
   DIGIPOT
1190 // #define DIGIPOT_I2C_ADDRESS_B 0x2D // Unshifted slave address for
   second DIGIPOT
1191 #endif
1192
1193 //=====
1194 =
1194 //=====Additional
1195 Features=====
1196 =
1197 // @section lcd
1198
1199 #if ANY(HAS_LCD_MENU, EXTENSIBLE_UI, HAS_DWIN_E3V2)
1200   #define MANUAL_FEEDRATE { 50*60, 50*60, 4*60, 2*60 } // (mm/min) Feedrates
   for manual moves along X, Y, Z, E from panel
1201   #define FINE_MANUAL_MOVE 0.025 // (mm) Smallest manual move (< 0.1mm)
   applying to Z on most machines
1202   #if IS_ULTIPANEL
1203     #define MANUAL_E_MOVES_RELATIVE // Display extruder move distance rather
   than "position"
1204     #define ULTIPANEL_FEEDMULTIPLY // Encoder sets the feedrate multiplier
   on the Status Screen
1205   #endif
1206 #endif
1207
1208 // Change values more rapidly when the encoder is rotated faster
1209 #define ENCODER_RATE_MULTIPLIER
1210 #if ENABLED(ENCODER_RATE_MULTIPLIER)
1211   #define ENCODER_10X_STEPS_PER_SEC 30 // (steps/s) Encoder rate for 10x
   speed
1212   #define ENCODER_100X_STEPS_PER_SEC 80 // (steps/s) Encoder rate for 100x
   speed
1213 #endif
1214
1215 // Play a beep when the feedrate is changed from the Status Screen
1216 // #define BEEP_ON_FEEDRATE_CHANGE
1217 #if ENABLED(BEEP_ON_FEEDRATE_CHANGE)
1218   #define FEEDRATE_CHANGE_BEEP_DURATION 10
1219   #define FEEDRATE_CHANGE_BEEP_FREQUENCY 440
1220 #endif
1221
1222 #if HAS_LCD_MENU

```

```

1223 // HAS_LCD_MENU
1224 // Add Probe Z Offset calibration to the Z Probe Offsets menu
1225 #if HAS_BED_PROBE
1226 // #define PROBE_OFFSET_WIZARD
1227 #if ENABLED(PROBE_OFFSET_WIZARD)
1228 //
1229 // Enable to init the Probe Z-Offset when starting the Wizard.
1230 // Use a height slightly above the estimated nozzle-to-probe Z offset.
1231 // For example, with an offset of -5, consider a starting height of
1232 // -4.
1233 // #
1234 // #define PROBE_OFFSET_WIZARD_START_Z -4.0
1235
1236 // Set a convenient position to do the calibration (probing point and
1237 // nozzle/bed-distance)
1238 // #define PROBE_OFFSET_WIZARD_XY_POS { X_CENTER, Y_CENTER }
1239 #endif
1240 #endif

1241 // Include a page of printer information in the LCD Main Menu
1242 // #define LCD_INFO_MENU
1243 #if ENABLED(LCD_INFO_MENU)
1244 // #define LCD_PRINTER_INFO_IS_BOOTSCREEN // Show bootscreen(s) instead
1245 // of Printer Info pages
1246 #endif

1247 // BACK menu items keep the highlight at the top
1248 // #define TURBO_BACK_MENU_ITEM

1249 // Add a mute option to the LCD menu
1250 // #define SOUND_MENU_ITEM

1251

1252 /**
1253 * LED Control Menu
1254 * Add LED Control to the LCD menu
1255 */
1256 // #define LED_CONTROL_MENU
1257 #if ENABLED(LED_CONTROL_MENU)
1258 #define LED_COLOR_PRESETS
1259 menu option
1260 // #define NE02_COLOR_PRESETS
1261 Preset Color menu option
1262 #if ENABLED(LED_COLOR_PRESETS)
1263 #define LED_USER_PRESET_RED 255 // User defined RED value
1264 #define LED_USER_PRESET_GREEN 128 // User defined GREEN value
1265 #define LED_USER_PRESET_BLUE 0 // User defined BLUE value
1266 #define LED_USER_PRESET_WHITE 255 // User defined WHITE value
1267 #define LED_USER_PRESET_BRIGHTNESS 255 // User defined intensity
1268 // #define LED_USER_PRESET_STARTUP // Have the printer display
1269 the user preset color on startup
1270 #endif
1271 #if ENABLED(NE02_COLOR_PRESETS)
1272 #define NE02_USER_PRESET_RED 255 // User defined RED value
1273 #define NE02_USER_PRESET_GREEN 128 // User defined GREEN value
1274 #define NE02_USER_PRESET_BLUE 0 // User defined BLUE value
1275 #define NE02_USER_PRESET_WHITE 255 // User defined WHITE value
1276 #define NE02_USER_PRESET_BRIGHTNESS 255 // User defined intensity
1277 // #define NE02_USER_PRESET_STARTUP // Have the printer display

```

```

the user preset color on startup for the second strip
1275 #endif
1276 #endif
1277
1278 // Insert a menu for preheating at the top level to allow for quick access
1279 // #define PREHEAT_SHORTCUT_MENU_ITEM
1280
1281 #endif // HAS_LCD_MENU
1282
1283 #if HAS_DISPLAY
1284     // The timeout (in ms) to return to the status screen from sub-menus
1285 // #define LCD_TIMEOUT_TO_STATUS 15000
1286
1287 #if ENABLED(SHOW_BOOTSCREEN)
1288     #define BOOTSCREEN_TIMEOUT 4000          // (ms) Total Duration to display
1289     the boot screen(s)
1290         #if EITHER(HAS_MARLINUI_U8GLIB, TFT_COLOR_UI)
1291             #define BOOT_MARLIN_LOGO_SMALL      // Show a smaller Marlin logo on
1292             the Boot Screen (saving lots of flash)
1293         #endif
1294     #endif
1295
1296 // Scroll a longer status message into view
1297 // #define STATUS_MESSAGE_SCROLLING
1298
1299 // On the Info Screen, display XY with one decimal place when possible
1300 // #define LCD_DECIMAL_SMALL_XY
1301
1302 // Add an 'M73' G-code to set the current percentage
1303 // #define LCD_SET_PROGRESS_MANUALLY
1304
1305 // Show the E position (filament used) during printing
1306 // #define LCD_SHOW_E_TOTAL
1307 #endif
1308
1309 // LCD Print Progress options
1310 #if EITHER(SDSUPPORT, LCD_SET_PROGRESS_MANUALLY)
1311     #if ANY(HAS_MARLINUI_U8GLIB, EXTENSIBLE_UI, HAS_MARLINUI_HD44780,
1312     IS_TFTLCD_PANEL, IS_DWIN_MARLINUI)
1313         // #define SHOW_REMAINING_TIME           // Display estimated time to
1314         completion
1315         #if ENABLED(SHOW_REMAINING_TIME)
1316             // #define USE_M73_REMAINING_TIME    // Use remaining time from M73
1317             command instead of estimation
1318             // #define ROTATE_PROGRESS_DISPLAY // Display (P)rogress, (E)lapsed,
1319             and (R)emaining time
1320         #endif
1321     #endif
1322
1323     #if EITHER(HAS_MARLINUI_U8GLIB, EXTENSIBLE_UI)
1324         // #define PRINT_PROGRESS_SHOW_DECIMALS // Show progress with decimal
1325         digits
1326     #endif
1327
1328     #if EITHER(HAS_MARLINUI_HD44780, IS_TFTLCD_PANEL)
1329         // #define LCD_PROGRESS_BAR          // Show a progress bar on HD44780
1330         LCDs for SD printing
1331         #if ENABLED(LCD_PROGRESS_BAR)
1332             #define DDNCDPEEE_RAD_RAD_TTME 2000 // (ms) Amount of time to show the

```

```

1324      #define PROGRESS_BAR_MSG_TIME 2000 // (ms) Amount of time to show the
bar
1325      #define PROGRESS_BAR_MSG_TIME 3000 // (ms) Amount of time to show the
status message
1326      #define PROGRESS_MSG_EXPIRE  0    // (ms) Amount of time to retain
the status message (0=forever)
1327          //##define PROGRESS_MSG_ONCE           // Show the message for MSG_TIME
then clear it
1328          //##define LCD_PROGRESS_BAR_TEST        // Add a menu item to test the
progress bar
1329      #endif
1330  #endif
1331 #endif
1332
1333 #if ENABLED(SDSUPPORT)
1334 /**
1335  * SD Card SPI Speed
1336  * May be required to resolve "volume init" errors.
1337  *
1338  * Enable and set to SPI_HALF_SPEED, SPI_QUARTER_SPEED, or
1339  * SPI_EIGHTH_SPEED
1340  * otherwise full speed will be applied.
1341  *
1342  * :['SPI_HALF_SPEED', 'SPI_QUARTER_SPEED', 'SPI_EIGHTH_SPEED']
1343 //##define SD_SPI_SPEED SPI_HALF_SPEED
1344
1345 // The standard SD detect circuit reads LOW when media is inserted and
1346 HIGH when empty.
1347 // Enable this option and set to HIGH if your SD cards are incorrectly
detected.
1348 //##define SD_DETECT_STATE HIGH
1349
1350 //##define SD_IGNORE_AT_STARTUP           // Don't mount the SD card when
starting up
1351 //##define SDCARD_READONLY               // Read-only SD card (to save
over 2K of flash)
1352
1353 //##define GCODE_REPEAT_MARKERS         // Enable G-code M808 to set
repeat markers and do looping
1354
1355 #define SD_PROCEDURE_DEPTH 1           // Increase if you need more
nested M32 calls
1356
1357 #define SD_FINISHED_STEPPERRELEASE true // Disable steppers when SD
Print is finished
1358 #define SD_FINISHED_RELEASECOMMAND "M84" // Use "M84XYE" to keep Z
enabled so your bed stays in place
1359
1360 // Reverse SD sort to show "more recent" files first, according to the
card's FAT.
1361 // Since the FAT gets out of order with usage, SDCARD_SORT_ALPHA is
recommended.
1362 #define SDCARD_RATHERRECENTFIRST
1363
1364 #define SD_MENU_CONFIRM_START          // Confirm the selected SD file
before printing
1365 //##define NO_SD_AUTOSTART           // Remove auto#.g file support

```

```

completely to save some Flash, SRAM
1366 // #define MENU_ADDAUTOSTART           // Add a menu option to run
auto#.g files

1367
1368 // #define BROWSE_MEDIA_ON_INSERT      // Open the file browser when
media is inserted

1369
1370 // #define MEDIA_MENU_AT_TOP          // Force the media menu to be
listed on the top of the main menu

1371
1372 #define EVENT_GCODE_SD_ABORT "G28XY"    // G-code to run on SD Abort
Print (e.g., "G28XY" or "G27")

1373
1374 #if ENABLED(PRINTER_EVENT_LEDS)
1375     #define PE_LEDS_COMPLETED_TIME (30*60) // (seconds) Time to keep the
LED "done" color before restoring normal illumination
1376 #endif

1377
1378 /**
1379 * Continue after Power-Loss (Creality3D)
1380 *
1381 * Store the current state to the SD Card at the start of each layer
1382 * during SD printing. If the recovery file is found at boot time, present
1383 * an option on the LCD screen to continue the print from the last-known
1384 * point in the file.
1385 */
1386 // #define POWER_LOSS_RECOVERY
1387 #if ENABLED(POWER_LOSS_RECOVERY)
1388     #define PLR_ENABLED_DEFAULT false // Power Loss Recovery enabled by
default. (Set with 'M413 Sn' & M500)
1389     // #define BACKUP_POWER_SUPPLY           // Backup power / UPS to move the
steppers on power loss
1390     // #define POWER_LOSS_ZRAISE            2 // (mm) Z axis raise on resume (on
power loss with UPS)
1391     // #define POWER_LOSS_PIN               44 // Pin to detect power loss. Set to
-1 to disable default pin on boards without module.
1392     // #define POWER_LOSS_STATE             HIGH // State of pin indicating power
loss
1393     // #define POWER_LOSS_PULLUP           // Set pullup / pulldown as
appropriate for your sensor
1394     // #define POWER_LOSS_PULLDOWN
1395     // #define POWER_LOSS_PURGE_LEN        20 // (mm) Length of filament to purge
on resume
1396     // #define POWER_LOSS_RETRACT_LEN       10 // (mm) Length of filament to
retract on fail. Requires backup power.

1397
1398     // Without a POWER_LOSS_PIN the following option helps reduce wear on
the SD card,
1399     // especially with "vase mode" printing. Set too high and vases cannot
be continued.
1400     #define POWER_LOSS_MIN_Z_CHANGE 0.05 // (mm) Minimum Z change before
saving power-loss data

1401
1402     // Enable if Z homing is needed for proper recovery. 99.9% of the time
this should be disabled!
1403     // #define POWER_LOSS_RECOVER_ZHOME
1404     #if ENABLED(POWER_LOSS_RECOVER_ZHOME)
1405         // #define POWER_LOSS_ZHOME_POS { 0, 0 } // Safe XY position to home Z
while avoiding objects on the bed

```

```

        WHILE COPYING OBJECTS ON THE SDU
1406    #endif
1407 #endif
1408
1409 /**
1410 * Sort SD file listings in alphabetical order.
1411 *
1412 * With this option enabled, items on SD cards will be sorted
1413 * by name for easier navigation.
1414 *
1415 * By default...
1416 *
1417 * - Use the slowest -but safest- method for sorting.
1418 * - Folders are sorted to the top.
1419 * - The sort key is statically allocated.
1420 * - No added G-code (M34) support.
1421 * - 40 item sorting limit. (Items after the first 40 are unsorted.)
1422 *
1423 * SD sorting uses static allocation (as set by SDSORT_LIMIT), allowing
the
1424 * compiler to calculate the worst-case usage and throw an error if the
SRAM
1425 * limit is exceeded.
1426 *
1427 * - SDSORT_USES_RAM provides faster sorting via a static directory
buffer.
1428 * - SDSORT_USES_STACK does the same, but uses a local stack-based
buffer.
1429 * - SDSORT_CACHE_NAMES will retain the sorted file listing in RAM.
(Expensive!)
1430 * - SDSORT_DYNAMIC_RAM only uses RAM when the SD menu is visible. (Use
with caution!)
1431 */
1432 // #define SDCARD_SORT_ALPHA
1433
1434 // SD Card Sorting options
1435 #if ENABLED(SDCARD_SORT_ALPHA)
1436     #define SDSORT_LIMIT          40      // Maximum number of sorted items (10-
256). Costs 27 bytes each.
1437     #define FOLDER_SORTING       -1      // -1=above  0=none  1=below
1438     #define SDSORT_GCODE         false   // Allow turning sorting on/off with
LCD and M34 G-code.
1439     #define SDSORT_USES_RAM       false   // Pre-allocate a static array for
faster pre-sorting.
1440     #define SDSORT_USES_STACK     false   // Prefer the stack for pre-sorting to
give back some SRAM. (Negated by next 2 options.)
1441     #define SDSORT_CACHE_NAMES   false   // Keep sorted items in RAM longer for
speedy performance. Most expensive option.
1442     #define SDSORT_DYNAMIC_RAM  false   // Use dynamic allocation (within SD
menus). Least expensive option. Set SDSORT_LIMIT before use!
1443     #define SDSORT_CACHE_VFATS  2       // Maximum number of 13-byte VFAT
entries to use for sorting.
1444                                         // Note: Only affects
SCROLL_LONG_Filenames with SDSORT_CACHE_NAMES but not SDSORT_DYNAMIC_RAM.
1445 #endif
1446
1447 // Allow international symbols in long filenames. To display correctly,
the
1448 // LCD's font must contain the characters. Check your selected LCD

```

```
language.
1449 // #define UTF_FILENAME_SUPPORT
1450
1451 // This allows hosts to request long names for files and folders with M33
1452 // #define LONG_FILENAME_HOST_SUPPORT
1453
1454 // Enable this option to scroll long filenames in the SD card menu
1455 // #define SCROLL_LONG_Filenames
1456
1457 // Leave the heaters on after Stop Print (not recommended!)
1458 // #define SD_ABORT_NO_COOLDOWN
1459
1460 /**
1461 * This option allows you to abort SD printing when any endstop is
1462 triggered.
1463 * This feature must be enabled with "M540 S1" or from the LCD menu.
1464 * To have any effect, endstops must be enabled during SD printing.
1465 */
1466 // #define SD_ABORT_ON_ENDSTOP_HIT
1467
1468 /**
1469 * This option makes it easier to print the same SD Card file again.
1470 * On print completion the LCD Menu will open with the file selected.
1471 * You can just click to start the print, or navigate elsewhere.
1472 */
1473 // #define SD_REPRINT_LAST_SELECTED_FILE
1474
1475 /**
1476 * Auto-report SdCard status with M27 S<seconds>
1477 */
1478 // #define AUTO_REPORT_SD_STATUS
1479
1480 /**
1481 * Support for USB thumb drives using an Arduino USB Host Shield or
1482 * equivalent MAX3421E breakout board. The USB thumb drive will appear
1483 * to Marlin as an SD card.
1484 *
1485 * The MAX3421E can be assigned the same pins as the SD card reader, with
1486 * the following pin mapping:
1487 *
1488 *      SCLK, MOSI, MISO --> SCLK, MOSI, MISO
1489 *      INT                 --> SD_DETECT_PIN [1]
1490 *      SS                  --> SDSS
1491 *
1492 * [1] On AVR an interrupt-capable pin is best for UHS3 compatibility.
1493 */
1494 // #define USB_FLASH_DRIVE_SUPPORT
1495 #if ENABLED(USB_FLASH_DRIVE_SUPPORT)
1496 /**
1497 * USB Host Shield Library
1498 *
1499 * - UHS2 uses no interrupts and has been production-tested
1500 *   on a LulzBot TAZ Pro with a 32-bit Archim board.
1501 *
1502 * - UHS3 is newer code with better USB compatibility. But it
1503 *   is less tested and is known to interfere with Servos.
1504 *   [1] This requires USB_INTR_PIN to be interrupt-capable.
1505 */
1506 // #define I2C_IIC_I2C_IICD
```

```

//#define USE_UHS3_USB
//#define USE_OTG_USB_HOST

/***
 * Native USB Host supported by some boards (USB OTG)
 */
//#define USE_OTG_USB_HOST

#if DISABLED(USE_OTG_USB_HOST)
#define USB_CS_PIN    SDSS
#define USB_INTR_PIN  SD_DETECT_PIN
#endif
#endif

/***
 * When using a bootloader that supports SD-Firmware-Flashing,
 * add a menu item to activate SD-FW-Update on the next reboot.
 *
 * Requires ATMEGA2560 (Arduino Mega)
 *
 * Tested with this bootloader:
 *   https://github.com/FleetProbe/MicroBridge-Arduino-ATMega2560
 */
#define SD_FIRMWARE_UPDATE
#if ENABLED(SD_FIRMWARE_UPDATE)
#define SD_FIRMWARE_UPDATE_EEPROM_ADDR      0x1FF
#define SD_FIRMWARE_UPDATE_ACTIVE_VALUE    0xF0
#define SD_FIRMWARE_UPDATE_INACTIVE_VALUE  0xFF
#endif

// Add an optimized binary file transfer mode, initiated with 'M28 B1'
#define BINARY_FILE_TRANSFER

/***
 * Set this option to one of the following (or the board's defaults
 * apply):
 *
 *      LCD - Use the SD drive in the external LCD controller.
 *      ONBOARD - Use the SD drive on the control board.
 *      CUSTOM_CABLE - Use a custom cable to access the SD (as defined in a
 * pins file).
 *
 *      * :[ 'LCD', 'ONBOARD', 'CUSTOM_CABLE' ]
 */
#define SDCARD_CONNECTION LCD

// Enable if SD detect is rendered useless (e.g., by using an SD extender)
#define NO_SD_DETECT

// Multiple volume support - EXPERIMENTAL.
#define MULTI_VOLUME
#if ENABLED(MULTI_VOLUME)
#define VOLUME_SD_ONBOARD
#define VOLUME_USB_FLASH_DRIVE
#define DEFAULT_VOLUME SV_SD_ONBOARD
#define DEFAULT_SHARED_VOLUME SV_USB_FLASH_DRIVE
#endif

#endif // SDSUPPORT

```

```
1562
1563 /**
1564 * By default an onboard SD card reader may be shared as a USB mass-
1565 * storage device. This option hides the SD card from the host PC.
1566 */
1567 // #define NO_SD_HOST_DRIVE // Disable SD Card access over USB (for
1568 security).
1569 /**
1570 * Additional options for Graphical Displays
1571 *
1572 * Use the optimizations here to improve printing performance,
1573 * which can be adversely affected by graphical display drawing,
1574 * especially when doing several short moves, and when printing
1575 * on DELTA and SCARA machines.
1576 *
1577 * Some of these options may result in the display lagging behind
1578 * controller events, as there is a trade-off between reliable
1579 * printing performance versus fast display updates.
1580 */
1581 #if HAS_MARLINUI_U8GLIB
1582     // Save many cycles by drawing a hollow frame or no frame on the Info
1583     Screen
1584     // #define XYZ_NO_FRAME
1585     #define XYZ_HOLLOW_FRAME
1586
1587     // A bigger font is available for edit items. Costs 3120 bytes of PROGMEM.
1588     // Western only. Not available for Cyrillic, Kana, Turkish, Greek, or
1589     Chinese.
1590     // #define USE_BIG_EDIT_FONT
1591
1592     // A smaller font may be used on the Info Screen. Costs 2434 bytes of
1593     PROGMEM.
1594     // Western only. Not available for Cyrillic, Kana, Turkish, Greek, or
1595     Chinese.
1596     // #define USE_SMALL_INFOFONT
1597
1598 /**
1599 * ST7920-based LCDs can emulate a 16 x 4 character display using
1600 * the ST7920 character-generator for very fast screen updates.
1601 * Enable LIGHTWEIGHT_UI to use this special display mode.
1602 *
1603 * Since LIGHTWEIGHT_UI has limited space, the position and status
1604 * message occupy the same line. Set STATUS_EXPIRE_SECONDS to the
1605 * length of time to display the status message before clearing.
1606 *
1607 * Set STATUS_EXPIRE_SECONDS to zero to never clear the status.
1608 * This will prevent position updates from being displayed.
1609 */
1610 #if ENABLED(U8GLIB_ST7920)
1611     // Enable this option and reduce the value to optimize screen updates.
1612     // The normal delay is 10µs. Use the lowest value that still gives a
1613     reliable display.
1614     // #define DOGM_SPI_DELAY_US 5
1615
1616     // #define LIGHTWEIGHT_UI
1617     #if ENABLED(LIGHTWEIGHT_UI)
1618         #define STATUS_EXPIRE_SECONDS 20
1619         #endif
1620
```

```

1014 #endif
1615 #endif
1616
1617 /**
1618 * Status (Info) Screen customizations
1619 * These options may affect code size and screen render time.
1620 * Custom status screens can forcibly override these settings.
1621 */
1622 //##define STATUS_COMBINE_HEATERS      // Use combined heater images instead
1623 of separate ones
1624 //##define STATUS_HOTEND_NUMBERLESS   // Use plain hotend icons instead of
1625 numbered ones (with 2+ hotends)
1626 #define STATUS_HOTEND_INVERTED       // Show solid nozzle bitmaps when
heating (Requires STATUS_HOTEND_ANIM for numbered hotends)
1627 #define STATUS_HOTEND_ANIM           // Use a second bitmap to indicate
hotend heating
1628 #define STATUS_BED_ANIM              // Use a second bitmap to indicate bed
heating
1629 #define STATUS_CHAMBER_ANIM         // Use a second bitmap to indicate
chamber heating
1630 //##define STATUS_CUTTER_ANIM        // Use a second bitmap to indicate
spindle / laser active
1631 //##define STATUS_COOLER_ANIM        // Use a second bitmap to indicate
laser cooling
1632 //##define STATUS_FLOWMETER_ANIM     // Use multiple bitmaps to indicate
coolant flow
1633 //##define STATUS_ALT_BED_BITMAP    // Use the alternative bed bitmap
1634 //##define STATUS_ALT_FAN_BITMAP     // Use the alternative fan bitmap
1635 //##define STATUS_FAN_FRAMES 3        // :[0,1,2,3,4] Number of fan
1636 animation frames
1637 // Frivolous Game Options
1638 //##define MARLIN_BRICKOUT
1639 //##define MARLIN_INVADERS
1640 //##define MARLIN_SNAKE
1641 //##define GAMES_EASTER_EGG          // Add extra blank lines above the
1642 "Games" sub-menu
1643#endif // HAS_MARLINUI_U8GLIB
1644
1645#if HAS_MARLINUI_U8GLIB || IS_DWIN_MARLINUI
1646 // Show SD percentage next to the progress bar
1647 //##define SHOW_SD_PERCENT
1648
1649 // Enable to save many cycles by drawing a hollow frame on Menu Screens
1650#define MENU_HOLLOW_FRAME
1651
1652 // Swap the CW/CCW indicators in the graphics overlay
1653 //##define OVERLAY_GFX_REVERSE
1654#endif
1655
1656//
1657// Additional options for DGUS / DWIN displays
1658//
1659#if HAS_DGUS_LCD
1660#define LCD_SERIAL_PORT 3

```

```

1661 #define LCD_BAUDRATE 115200
1662
1663 #define DGUS_RX_BUFFER_SIZE 128
1664 #define DGUS_TX_BUFFER_SIZE 48
1665 // #define SERIAL_STATS_RX_BUFFER_OVERRUNS // Fix Rx overrun situation
1666 // (Currently only for AVR)
1667
1668 #define DGUS_UPDATE_INTERVAL_MS 500 // (ms) Interval between automatic
1669 screen updates
1670
1671 #if ANY(DGUS_LCD_UI_FYSETC, DGUS_LCD_UI_MKS, DGUS_LCD_UI_HIPRECY)
1672     #define DGUS_PRINT_FILENAME // Display the filename during
1673 printing
1674     #define DGUS_PREHEAT_UI // Display a preheat screen during
1675 heatup
1676
1677 #if EITHER(DGUS_LCD_UI_FYSETC, DGUS_LCD_UI_MKS)
1678     // #define DGUS_UI_MOVE_DIS_OPTION // Disabled by default for FYSETC
1679 and MKS
1680     #else
1681         #define DGUS_UI_MOVE_DIS_OPTION // Enabled by default for
1682 UI_HIPRECY
1683     #endif
1684
1685 #define DGUS_FILAMENT_LOADUNLOAD
1686 #if ENABLED(DGUS_FILAMENT_LOADUNLOAD)
1687     #define DGUS_FILAMENT_PURGE_LENGTH 10
1688     #define DGUS_FILAMENT_LOAD_LENGTH_PER_TIME 0.5 // (mm) Adjust in
1689 proportion to DGUS_UPDATE_INTERVAL_MS
1690 #endif
1691
1692 #define DGUS_UI_WAITING // Show a "waiting" screen between
1693 some screens
1694 #if ENABLED(DGUS_UI_WAITING)
1695     #define DGUS_UI_WAITING_STATUS 10
1696     #define DGUS_UI_WAITING_STATUS_PERIOD 8 // Increase to slower waiting
1697 status looping
1698 #endif
1699 #endif // HAS_DGUS_LCD
1700
1701 // Additional options for AnyCubic Chiron TFT displays
1702 //
1703 // #if ENABLED(ANYCUBIC_LCD_CHIRON)
1704 // By default the type of panel is automatically detected.
1705 // Enable one of these options if you know the panel type.
1706 // #define CHIRON_TFT_STANDARD
1707 // #define CHIRON_TFT_NEW
1708
1709 // Enable the longer Anycubic powerup startup tune
1710 // #define AC_DEFAULT_STARTUP_TUNE
1711
1712 /**
1713 * Display Folders
1714 * By default the file browser lists all G-code files (including those in
1715 subfolders) in a flat list.
1716 * Enable this option to display a hierarchical file browser.
1717 */

```

```

1 / 09 *
1710 * NOTES:
1711 * - Without this option it helps to enable SDCARD_SORT_ALPHA so files are
1712 sorted before/after folders.
1713 * - When used with the "new" panel, folder names will also have '.gcode'
1714 appended to their names.
1715 * This hack is currently required to force the panel to show folders.
1716 */
1717 #define AC_SD_FOLDER_VIEW
1718 #endif
1719 // 
1720 // Specify additional languages for the UI. Default specified by
1721 LCD_LANGUAGE.
1722 //
1723 #if ANY(DOGLCD, TFT_COLOR_UI, TOUCH_UI_FTDI_EVE, IS_DWIN_MARLINUI)
1724     //#define LCD_LANGUAGE_2 fr
1725     //#define LCD_LANGUAGE_3 de
1726     //#define LCD_LANGUAGE_4 es
1727     //#define LCD_LANGUAGE_5 it
1728     #ifdef LCD_LANGUAGE_2
1729         //#define LCD_LANGUAGE_AUTO_SAVE // Automatically save language to
1730 EEPROM on change
1731     #endif
1732 #endif
1733 // 
1734 // Touch UI for the FTDI Embedded Video Engine (EVE)
1735 //
1736 #if ENABLED(TOUCH_UI_FTDI_EVE)
1737     // Display board used
1738     //#define LCD_FTDI_VM800B35A      // FTDI 3.5" with FT800 (320x240)
1739     //#define LCD_4DSYSTEMS_4LCD_FT843 // 4D Systems 4.3" (480x272)
1740     //#define LCD_HAOYU_FT800CB      // Haoyu with 4.3" or 5" (480x272)
1741     //#define LCD_HAOYU_FT810CB      // Haoyu with 5" (800x480)
1742     //#define LCD_LULZBOT_CLCD_UI    // LulzBot Color LCD UI
1743     //#define LCD_FYSETC_TFT81050    // FYSETC with 5" (800x480)
1744     //#define LCD_EVE3_50G          // Matrix Orbital 5.0", 800x480, BT815
1745     //#define LCD_EVE2_50G          // Matrix Orbital 5.0", 800x480, FT813
1746 // Correct the resolution if not using the stock TFT panel.
1747     //#define TOUCH_UI_320x240
1748     //#define TOUCH_UI_480x272
1749     //#define TOUCH_UI_800x480
1750 // Mappings for boards with a standard RepRapDiscount Display connector
1751     //#define A0_EXP1_PINMAP        // LulzBot CLCD UI EXP1 mapping
1752     //#define A0_EXP2_PINMAP        // LulzBot CLCD UI EXP2 mapping
1753     //#define CR10_TFT_PINMAP       // Rudolph Riedel's CR10 pin mapping
1754     //#define S6_TFT_PINMAP         // FYSETC S6 pin mapping
1755     //#define F6_TFT_PINMAP         // FYSETC F6 pin mapping
1756 // #define OTHER_PIN_LAYOUT // Define pins manually below
1757 #if ENABLED(OTHER_PIN_LAYOUT)
1758     // Pins for CS and MOD_RESET (PD) must be chosen
1759     #define CLCD_MOD_RESET 9
1760     #define CLCD_SPI_CS    10
1761 // If using software SPI, specify pins for SCLK, MOSI, MISO
1762
1763

```

```

1764 // #define CLCD_USE_SOFT_SPI
1765 #if ENABLED(CLCD_USE_SOFT_SPI)
1766     #define CLCD_SOFT_SPI_MOSI 11
1767     #define CLCD_SOFT_SPI_MISO 12
1768     #define CLCD_SOFT_SPI_SCLK 13
1769 #endif
1770#endif
1771
1772 // Display Orientation. An inverted (i.e. upside-down) display
1773 // is supported on the FT800. The FT810 and beyond also support
1774 // portrait and mirrored orientations.
1775 // #define TOUCH_UI_INVERTED
1776 // #define TOUCH_UI_PORTRAIT
1777 // #define TOUCH_UI_MIRRORED
1778
1779 // UTF8 processing and rendering.
1780 // Unsupported characters are shown as '?'.
1781 // #define TOUCH_UI_USE_UTF8
1782 #if ENABLED(TOUCH_UI_USE_UTF8)
1783     // Western accents support. These accented characters use
1784     // combined bitmaps and require relatively little storage.
1785     #define TOUCH_UI_UTF8_WESTERN_CHARSET
1786 #if ENABLED(TOUCH_UI_UTF8_WESTERN_CHARSET)
1787     // Additional character groups. These characters require
1788     // full bitmaps and take up considerable storage:
1789     // #define TOUCH_UI_UTF8_SUPERSCRIPTS // ¹²³
1790     // #define TOUCH_UI_UTF8_COPYRIGHT // © ®
1791     // #define TOUCH_UI_UTF8_GERMANIC // ß
1792     // #define TOUCH_UI_UTF8_SCANDINAVIAN // æ ð ø þ æ ð ø þ
1793     // #define TOUCH_UI_UTF8_PUNCTUATION // « » ¿ ¡
1794     // #define TOUCH_UI_UTF8_CURRENCY // ¢ £ ¤ ¥
1795     // #define TOUCH_UI_UTF8_ORDINALS // ° ª
1796     // #define TOUCH_UI_UTF8_MATHEMATICS // ± × ÷
1797     // #define TOUCH_UI_UTF8_FRACTIONS // ¼ ½ ¾
1798     // #define TOUCH_UI_UTF8_SYMBOLS // µ ¶ ¦ § ¬
1799 #endif
1800
1801 // Cyrillic character set, costs about 27KiB of flash
1802 // #define TOUCH_UI_UTF8_CYRILLIC_CHARSET
1803#endif
1804
1805 // Use a smaller font when labels don't fit buttons
1806 #define TOUCH_UI_FIT_TEXT
1807
1808 // Use a numeric passcode for "Screen lock" keypad.
1809 // (recommended for smaller displays)
1810 // #define TOUCH_UI_PASSCODE
1811
1812 // Output extra debug info for Touch UI events
1813 // #define TOUCH_UI_DEBUG
1814
1815 // Developer menu (accessed by touching "About Printer" copyright text)
1816 // #define TOUCH_UI_DEVELOPER_MENU
1817#endif
1818
1819 //
1820 // Classic UI Options
1821 //
1822 // TTF SCALING MODE

```

```

1824 #if TFT_SCALED_DUGLCD
1825     //#define TFT_MARLINUI_COLOR 0xFFFF // White
1826     //#define TFT_MARLINBG_COLOR 0x0000 // Black
1827     //#define TFT_DISABLED_COLOR 0x0003 // Almost black
1828     //#define TFT_BTCANCEL_COLOR 0xF800 // Red
1829     //#define TFT_BTARROWS_COLOR 0xDEE6 // 110111 110111 00110 Yellow
1830     //#define TFT_BTOKMENU_COLOR 0x145F // 00010 100010 11111 Cyan
1831 #endif
1832 //
1833 // ADC Button Debounce
1834 //
1835 #if HAS_ADC_BUTTONS
1836     #define ADC_BUTTON_DEBOUNCE_DELAY 16 // Increase if buttons bounce or
1837     repeat too fast
1838 #endif
1839 //
1840 // @section safety
1841 /**
1842 * The watchdog hardware timer will do a reset and disable all outputs
1843 * if the firmware gets too overloaded to read the temperature sensors.
1844 *
1845 * If you find that watchdog reboot causes your AVR board to hang forever,
1846 * enable WATCHDOG_RESET_MANUAL to use a custom timer instead of WDTO.
1847 * NOTE: This method is less reliable as it can only catch hangups while
1848 * interrupts are enabled.
1849 */
1850 #define USE_WATCHDOG
1851 #if ENABLED(USE_WATCHDOG)
1852     //#define WATCHDOG_RESET_MANUAL
1853 #endif
1854 //
1855 // @section lcd
1856 /**
1857 * Babystepping enables movement of the axes by tiny increments without
1858 * changing
1859 * the current position values. This feature is used primarily to adjust the
1860 * Z
1861 * axis in the first layer of a print in real-time.
1862 *
1863 * Warning: Does not respect endstops!
1864 */
1865 //#define BABYSTEPPING
1866 #if ENABLED(BABYSTEPPING)
1867     //#define INTEGRATED_BABYSTEPPING          // EXPERIMENTAL integration of
1868     babystepping into the Stepper ISR
1869     //#define BABYSTEP_WITHOUT_HOMING         // Allow babystepping at all
1870     //#define BABYSTEP_ALWAYS_AVAILABLE       times (not just during movement).
1871     //#define BABYSTEP_XY                   // Also enable X/Y Babystepping.
1872     Not supported on DELTA!
1873     #define BABYSTEP_INVERT_Z false        // Change if Z babysteps should
1874     go the other way
1875     //#define BABYSTEP_MILLIMETER_UNITS      // Specify
1876     BABYSTEP_MULTIPLICATOR_(XY|Z) in mm instead of micro-steps
1877     #define BABYSTEP_MULTIPLICATOR_Z 1      // (steps or mm) Steps or
1878     millimeter distance for each Z babystep

```

```

1872 #define BABYSTEP_MULTIPLICATOR_XY 1      // (steps or mm) Steps or
millimeter distance for each XY babystep
1873
1874 //##define DOUBLECLICK_FOR_Z_BABYSTEPPING // Double-click on the Status
Screen for Z Babystepping.
1875 #if ENABLED(DOUBLECLICK_FOR_Z_BABYSTEPPING)
1876     #define DOUBLECLICK_MAX_INTERVAL 1250 // Maximum interval between
clicks, in milliseconds.
1877                                         // Note: Extra time may be added
to mitigate controller latency.
1878     //##define MOVE_Z_WHEN_IDLE           // Jump to the move Z menu on
doubleclick when printer is idle.
1879     #if ENABLED(MOVE_Z_WHEN_IDLE)
1880         #define MOVE_Z_IDLE_MULTIPLICATOR 1 // Multiply 1mm by this factor
for the move step size.
1881     #endif
1882 #endif
1883
1884 //##define BABYSTEP_DISPLAY_TOTAL          // Display total babysteps since
last G28
1885
1886 //##define BABYSTEP_ZPROBE_OFFSET          // Combine M851 Z and
Babystepping
1887 #if ENABLED(BABYSTEP_ZPROBE_OFFSET)
1888     //##define BABYSTEP_HOTEND_Z_OFFSET      // For multiple hotends,
babystep relative Z offsets
1889     //##define BABYSTEP_ZPROBE_GFX_OVERLAY    // Enable graphical overlay on
Z-offset editor
1890 #endif
1891#endif
1892
1893 // @section extruder
1894
1895 /**
 * Linear Pressure Control v1.5
 *
 * Assumption: advance [steps] = k * (delta velocity [steps/s])
 * K=0 means advance disabled.
 *
 * NOTE: K values for LIN_ADVANCE 1.5 differ from earlier versions!
 *
 * Set K around 0.22 for 3mm PLA Direct Drive with ~6.5cm between the drive
gear and heatbreak.
 * Larger K values will be needed for flexible filament and greater
distances.
 * If this algorithm produces a higher speed offset than the extruder can
handle (compared to E jerk)
 * print acceleration will be reduced during the affected moves to keep
within the limit.
 *
 * See https://marlinfw.org/docs/features/lin\_advance.html for full
instructions.
 */
1910 //##define LIN_ADVANCE
1911 #if ENABLED(LIN_ADVANCE)
1912     //##define EXTRA_LIN_ADVANCE_K // Enable for second linear advance
constants
1913     #define LIN_ADVANCE_K 0.22 // Unit: mm compression per 1mm/s extruder

```

```

speed
  //#define LA_DEBUG           // If enabled, this will generate debug
information output over USB.
  //#define EXPERIMENTAL_SCURVE // Enable this option to permit S-Curve
Acceleration
#endif

// @section leveling

/***
 * Points to probe for all 3-point Leveling procedures.
 * Override if the automatically selected points are inadequate.
 */
#ifndef EITHER(AUTO_BED_LEVELING_3POINT, AUTO_BED_LEVELINGUBL)
  //#define PROBE_PT_1_X 15
  //#define PROBE_PT_1_Y 180
  //#define PROBE_PT_2_X 15
  //#define PROBE_PT_2_Y 20
  //#define PROBE_PT_3_X 170
  //#define PROBE_PT_3_Y 20
#endif

/***
 * Probing Margins
 *
 * Override PROBING_MARGIN for each side of the build plate
 * Useful to get probe points to exact positions on targets or
 * to allow leveling to avoid plate clamps on only specific
 * sides of the bed. With NOZZLE_AS_PROBE negative values are
 * allowed, to permit probing outside the bed.
 *
 * If you are replacing the prior *_PROBE_BED_POSITION options,
 * LEFT and FRONT values in most cases will map directly over
 * RIGHT and REAR would be the inverse such as
 * (X/Y_BED_SIZE - RIGHT/BACK_PROBE_BED_POSITION)
 *
 * This will allow all positions to match at compilation, however
 * should the probe position be modified with M851XY then the
 * probe points will follow. This prevents any change from causing
 * the probe to be unable to reach any points.
 */
#ifndef PROBE_SELECTED && !IS_KINEMATIC
  #define PROBING_MARGIN_LEFT PROBING_MARGIN
  #define PROBING_MARGIN_RIGHT PROBING_MARGIN
  #define PROBING_MARGIN_FRONT PROBING_MARGIN
  #define PROBING_MARGIN_BACK PROBING_MARGIN
#endif

#ifndef EITHER(MESH_BED_LEVELING, AUTO_BED_LEVELINGUBL)
  // Override the mesh area if the automatic (max) area is too large
  #define MESH_MIN_X MESH_INSET
  #define MESH_MIN_Y MESH_INSET
  #define MESH_MAX_X X_BED_SIZE - (MESH_INSET)
  #define MESH_MAX_Y Y_BED_SIZE - (MESH_INSET)
#endif

#ifndef BOTH(AUTO_BED_LEVELINGUBL, EEPROM_SETTINGS)
  #define OPTIMIZED_MESH_STORAGE // Store mesh with less precision to
save EEPROM space

```

```

1969 #endif
1970
1971 /**
1972 * Repeatedly attempt G29 leveling until it succeeds.
1973 * Stop after G29_MAX_RETRIES attempts.
1974 */
1975 // #define G29_RETRY_AND_RECOVER
1976 #if ENABLED(G29_RETRY_AND_RECOVER)
1977     #define G29_MAX_RETRIES 3
1978     #define G29_HALT_ON_FAILURE
1979 /**
1980 * Specify the GCODE commands that will be executed when leveling
1981 succeeds,
1982 * between attempts, and after the maximum number of retries have been
1983 tried.
1984 */
1985 #define G29_SUCCESS_COMMANDS "M117 Bed leveling done."
1986 #define G29_RECOVER_COMMANDS "M117 Probe failed. Rewiping.\nG28\nG12 P0
S12 T0"
1987 #define G29_FAILURE_COMMANDS "M117 Bed leveling failed.\nG0 Z10\nM300 P25
S880\nM300 P50 S0\nM300 P25 S880\nM300 P50 S0\nM300 P25 S880\nM300 P50
S0\nG4 S1"
1988
1989#endif
1990
1991 /**
1992 * Thermal Probe Compensation
1993 * Probe measurements are adjusted to compensate for temperature distortion.
1994 * Use G76 to calibrate this feature. Use M871 to set values manually.
1995 * For a more detailed explanation of the process see G76_M871.cpp.
1996 */
1997 #if HAS_BED_PROBE && TEMP_SENSOR_PROBE && TEMP_SENSOR_BED
1998     // Enable thermal first layer compensation using bed and probe
1999     // temperatures
2000     #define PROBE_TEMP_COMPENSATION
2001
2002         // Add additional compensation depending on hotend temperature
2003         // Note: this values cannot be calibrated and have to be set manually
2004         #if ENABLED(PROBE_TEMP_COMPENSATION)
2005             // Park position to wait for probe cooldown
2006             #define PTC_PARK_POS { 0, 0, 100 }
2007
2008             // Probe position to probe and wait for probe to reach target
2009             // temperature
2010             #define PTC_PROBE_POS { 90, 100 }
2011
2012             // Enable additional compensation using hotend temperature
2013             // Note: this values cannot be calibrated automatically but have to be
2014             // set manually
2015             // #define USE_TEMP_EXT_COMPENSATION
2016
2017             // Probe temperature calibration generates a table of values starting at
2018             // PTC_SAMPLE_START
2019             // (e.g., 30), in steps of PTC_SAMPLE_RES (e.g., 5) with
2020             // PTC_SAMPLE_COUNT (e.g., 10) samples.
2021
2022             // #define PTC_SAMPLE_START 30 // (°C)
2023             // #define PTC_SAMPLE_RES 5 // (°C)

```

```

2017 //##define PTC_SAMPLE_COUNT 10
2018 // Bed temperature calibration builds a similar table.
2019
2020 //##define BTC_SAMPLE_START 60 // (°C)
2021 //##define BTC_SAMPLE_RES 5 // (°C)
2022 //##define BTC_SAMPLE_COUNT 10
2023
2024 // The temperature the probe should be at while taking measurements
2025 during bed temperature
2026 // calibration.
2027 //##define BTC_PROBE_TEMP 30 // (°C)
2028
2029 // Height above Z=0.0 to raise the nozzle. Lowering this can help the
2030 probe to heat faster.
2031 // Note: the Z=0.0 offset is determined by the probe offset which can be
2032 set using M851.
2033 //##define PTC_PROBE_HEATING_OFFSET 0.5
2034
2035 // Height to raise the Z-probe between heating and taking the next
2036 measurement. Some probes
2037 // may fail to untrigger if they have been triggered for a long time,
2038 which can be solved by
2039 // increasing the height the probe is raised to.
2040 //##define PTC_PROBE_RAISE 15
2041
2042 // If the probe is outside of the defined range, use linear
2043 extrapolation using the closest
2044 // point and the PTC_LINEAR_EXTRAPOLATION'th next point. E.g. if set to
2045 4 it will use data[0]
2046 // and data[4] to perform linear extrapolation for values below
2047 PTC_SAMPLE_START.
2048 //##define PTC_LINEAR_EXTRAPOLATION 4
2049 #endif
2050#endif
2051
2052// @section extras
2053
2054// G60/G61 Position Save and Return
2055// G60/G61 Position Save and Return
2056//##define SAVED_POSITIONS 1 // Each saved position slot costs 12
2057 bytes
2058
2059// G2/G3 Arc Support
2060// G2/G3 Arc Support
2061#define ARC_SUPPORT // Requires ~3226 bytes
2062#if ENABLED(ARC_SUPPORT)
2063  #define MIN_ARC_SEGMENT_MM 0.1 // (mm) Minimum length of each arc
2064  segment
2065  #define MAX_ARC_SEGMENT_MM 1.0 // (mm) Maximum length of each arc
2066  segment
2067  #define MIN_CIRCLE_SEGMENTS 72 // Minimum number of segments in a
2068  complete circle
2069  //##define ARC_SEGMENTS_PER_SEC 50 // Use the feedrate to choose the
2070  segment length
2071  #define N_ARC_CORRECTION 25 // Number of interpolated segments
2072  between corrections

```

```
2062 //##define ARC_P_CIRCLES           // Enable the 'P' parameter to specify
2063 complete circles
2064 //##define SF_ARC_FIX             // Enable only if using SkeinForge
2065 with "Arc Point" fillet procedure
2066 #endif
2067
2068 // G5 Bézier Curve Support with XYZE destination and IJPQ offsets
2069 //##define BEZIER_CURVE_SUPPORT    // Requires ~2666 bytes
2070
2071 #if EITHER(ARC_SUPPORT, BEZIER_CURVE_SUPPORT)
2072     //##define CNC_WORKSPACE_PLANES   // Allow G2/G3/G5 to operate in XY,
2073 ZX, or YZ planes
2074 #endif
2075
2076 /**
2077 * Direct Stepping
2078 *
2079 * Comparable to the method used by Klipper, G6 direct stepping
2080 significantly
2081 * reduces motion calculations, increases top printing speeds, and results
2082 in
2083 * less step aliasing by calculating all motions in advance.
2084 * Preparing your G-code: https://github.com/colinrgodsey/step-daemon
2085 */
2086 //##define DIRECT_STEPPING
2087
2088 /**
2089 * G38 Probe Target
2090 *
2091 * This option adds G38.2 and G38.3 (probe towards target)
2092 * and optionally G38.4 and G38.5 (probe away from target).
2093 * Set MULTIPLE_PROBING for G38 to probe more than once.
2094 */
2095 //##define G38_PROBE_TARGET
2096 #if ENABLED(G38_PROBE_TARGET)
2097     //##define G38_PROBE_AWAY        // Include G38.4 and G38.5 to probe away
2098 from target
2099     #define G38_MINIMUM_MOVE 0.0275 // (mm) Minimum distance that will produce
2100 a move.
2101 #endif
2102
2103 // Moves (or segments) with fewer steps than this will be joined with the
2104 next move
2105 #define MIN_STEPS_PER_SEGMENT 6
2106
2107 /**
2108 * Minimum delay before and after setting the stepper DIR (in ns)
2109 * 0 : No delay (Expect at least 10µS since one Stepper ISR must
2110 transpire)
2111 * 20 : Minimum for TMC2xxx drivers
2112 * 200 : Minimum for A4988 drivers
2113 * 400 : Minimum for A5984 drivers
2114 * 500 : Minimum for LV8729 drivers (guess, no info in datasheet)
2115 * 650 : Minimum for DRV8825 drivers
2116 * 1500 : Minimum for TB6600 drivers (guess, no info in datasheet)
2117 * 15000 : Minimum for TB6560 drivers (guess, no info in datasheet)
2118 *
2119 * Override the default value based on the driver type set in
```

```

Configuration.h.

2111 */
2112 //#define MINIMUM_STEPPER_POST_DIR_DELAY 650
2113 //#define MINIMUM_STEPPER_PRE_DIR_DELAY 650
2114
2115 /**
2116 * Minimum stepper driver pulse width (in  $\mu$ s)
2117 * 0 : Smallest possible width the MCU can produce, compatible with
2118 TMC2xxx drivers
2119 * 0 : Minimum 500ns for LV8729, adjusted in stepper.h
2120 * 1 : Minimum for A4988 and A5984 stepper drivers
2121 * 2 : Minimum for DRV8825 stepper drivers
2122 * 3 : Minimum for TB6600 stepper drivers
2123 * 30 : Minimum for TB6560 stepper drivers
2124 *
2125 * Override the default value based on the driver type set in
2126 Configuration.h.
2127 */
2128 /**
2129 * Maximum stepping rate (in Hz) the stepper driver allows
2130 * If undefined, defaults to 1MHz / (2 * MINIMUM_STEPPER_PULSE)
2131 * 5000000 : Maximum for TMC2xxx stepper drivers
2132 * 1000000 : Maximum for LV8729 stepper driver
2133 * 500000 : Maximum for A4988 stepper driver
2134 * 250000 : Maximum for DRV8825 stepper driver
2135 * 150000 : Maximum for TB6600 stepper driver
2136 * 15000 : Maximum for TB6560 stepper driver
2137 *
2138 * Override the default value based on the driver type set in
2139 Configuration.h.
2140 */
2141 //#define MAXIMUM_STEPPER_RATE 250000

2142 // @section temperature
2143
2144 // Control heater 0 and heater 1 in parallel.
2145 //#define HEATERS_PARALLEL
2146
2147 //=====
2148 //=====
2149 //===== Buffers
2150 //=====
2151
2152 // @section motion
2153
2154 // The number of linear moves that can be in the planner at once.
2155 // The value of BLOCK_BUFFER_SIZE must be a power of 2 (e.g., 8, 16, 32)
2156 #if BOTH(SDSUPPORT, DIRECT_STEPPING)
2157     #define BLOCK_BUFFER_SIZE 8
2158 #elif ENABLED(SDSUPPORT)
2159     #define BLOCK_BUFFER_SIZE 16
2160 #else
2161     #define BLOCK_BUFFER_SIZE 16
2162 #endif

```

```

2163 // @section serial
2164
2165 // The ASCII buffer for serial input
2166 #define MAX_CMD_SIZE 96
2167 #define BUFSIZE 4
2168
2169 // Transmission to Host Buffer Size
2170 // To save 386 bytes of PROGMEM (and TX_BUFFER_SIZE+3 bytes of RAM) set to
2171 // 0.
2172 // To buffer a simple "ok" you need 4 bytes.
2173 // For ADVANCED_OK (M105) you need 32 bytes.
2174 // For debug-echo: 128 bytes for the optimal speed.
2175 // Other output doesn't need to be that speedy.
2176 // :[0, 2, 4, 8, 16, 32, 64, 128, 256]
2177 #define TX_BUFFER_SIZE 0
2178
2179 // Host Receive Buffer Size
2180 // Without XON/XOFF flow control (see SERIAL_XON_XOFF below) 32 bytes should
2181 // be enough.
2182 // To use flow control, set this buffer size to at least 1024 bytes.
2183 // :[0, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048]
2184 // #define RX_BUFFER_SIZE 1024
2185
2186 #if RX_BUFFER_SIZE >= 1024
2187     // Enable to have the controller send XON/XOFF control characters to
2188     // the host to signal the RX buffer is becoming full.
2189     // #define SERIAL_XON_XOFF
2190 #endif
2191
2192 #if ENABLED(SDSUPPORT)
2193     // Enable this option to collect and display the maximum
2194     // RX queue usage after transferring a file to SD.
2195     // #define SERIAL_STATS_MAX_RX_QUEUED
2196
2197     // Enable this option to collect and display the number
2198     // of dropped bytes after a file transfer to SD.
2199     // #define SERIAL_STATS_DROPPED_RX
2200 #endif
2201
2202 // Monitor RX buffer usage
2203 // Dump an error to the serial port if the serial receive buffer overflows.
2204 // If you see these errors, increase the RX_BUFFER_SIZE value.
2205 // Not supported on all platforms.
2206 // #define RX_BUFFER_MONITOR
2207
2208 /**
2209 * Emergency Command Parser
2210 *
2211 * Add a low-level parser to intercept certain commands as they
2212 * enter the serial receive buffer, so they cannot be blocked.
2213 * Currently handles M108, M112, M410, M876
2214 * NOTE: Not yet implemented for all platforms.
2215 */
2216 // #define EMERGENCY_PARSER
2217
2218 /**
2219 * Realtime Reporting (requires EMERGENCY_PARSER)
2220 */

```

```

2219 * - Report position and state of the machine (like Grbl).
2220 * - Auto-report position during long moves.
2221 * - Useful for CNC/LASER.
2222 *
2223 * Adds support for commands:
2224 * S000 : Report State and Position while moving.
2225 * P000 : Instant Pause / Hold while moving.
2226 * R000 : Resume from Pause / Hold.
2227 *
2228 * - During Hold all Emergency Parser commands are available, as usual.
2229 * - Enable NANODLP_Z_SYNC and NANODLP_ALL_AXIS for move command end-state
2230 reports.
2231 */
2232 //#define REALTIME_REPORTING_COMMANDS
2233 #if ENABLED(REALTIME_REPORTING_COMMANDS)
2234     //#define FULL_REPORT_TO_HOST_FEATURE // Auto-report the machine status
2235     // like Grbl CNC
2236 #endif
2237
2238 // Bad Serial-connections can miss a received command by sending an 'ok'
2239 // Therefore some clients abort after 30 seconds in a timeout.
2240 // Some other clients start sending commands while receiving a 'wait'.
2241 // This "wait" is only sent when the buffer is empty. 1 second is a good
2242 // value here.
2243 //#define NO_TIMEOUTS 1000 // Milliseconds
2244
2245 // Some clients will have this feature soon. This could make the NO_TIMEOUTS
2246 // unnecessary.
2247 //#define ADVANCED_OK
2248
2249 // Printronix may have trouble receiving long strings all at once.
2250 // This option inserts short delays between lines of serial output.
2251 #define SERIAL_OVERRUN_PROTECTION
2252
2253 // For serial echo, the number of digits after the decimal point
2254 //#define SERIAL_FLOAT_PRECISION 4
2255
2256 // @section extras
2257
2258 /**
2259 * Extra Fan Speed
2260 * Adds a secondary fan speed for each print-cooling fan.
2261 * 'M106 P<fan> T3-255' : Set a secondary speed for <fan>
2262 * 'M106 P<fan> T2'      : Use the set secondary speed
2263 * 'M106 P<fan> T1'      : Restore the previous fan speed
2264 */
2265 //#define EXTRA_FAN_SPEED
2266
2267 /**
2268 * Firmware-based and LCD-controlled retract
2269 *
2270 * Add G10 / G11 commands for automatic firmware-based retract / recover.
2271 * Use M207 and M208 to define parameters for retract / recover.
2272 *
2273 * Use M209 to enable or disable auto-retract.
2274 * With auto-retract enabled, all G1 E moves within the set range
2275 * will be converted to firmware-based retract/recover moves.
2276 *
2277 * Re-cure to turn off auto_retract during filament change

```

```

2273 * DC_SURE_TO_TURN_OFF_AUTO_RETRACT_DURING_TOOLCHANGE_CHANGE.
2274 *
2275 * Note that M207 / M208 / M209 settings are saved to EEPROM.
2276 */
2277 // #define FWRETRACT
2278 #if ENABLED(FWRETRACT)
2279     #define FWRETRACT_AUTORETRACT           // Override slicer retractions
2280         #if ENABLED(FWRETRACT_AUTORETRACT)
2281             #define MIN_AUTORETRACT          0.1 // (mm) Don't convert E moves
2282             under this length
2283             #define MAX_AUTORETRACT        10.0 // (mm) Don't convert E moves
2284             over this length
2285             #endif
2286             #define RETRACT_LENGTH          3    // (mm) Default retract length
2287             (positive value)
2288             #define RETRACT_LENGTH_SWAP     13   // (mm) Default swap retract
2289             length (positive value)
2290             #define RETRACT_FEEDRATE       45   // (mm/s) Default feedrate for
2291             retracting
2292             #define RETRACT_ZRAISE          0    // (mm) Default retract Z-raise
2293             #define RETRACT_COVER_LENGTH    0    // (mm) Default additional
2294             recover length (added to retract length on recover)
2295             #define RETRACT_COVER_LENGTH_SWAP 0    // (mm) Default additional swap
2296             recover length (added to retract length on recover from toolchange)
2297             #define RETRACT_COVER_FEEDRATE  8    // (mm/s) Default feedrate for
2298             recovering from retraction
2299             #define RETRACT_COVER_FEEDRATE_SWAP 8  // (mm/s) Default feedrate for
2300             recovering from swap retraction
2301             #if ENABLED(MIXING_EXTRUDER)
2302                 // #define RETRACT_SYNC_MIXING      // Retract and restore all
2303                 mixing steppers simultaneously
2304             #endif
2305         #endif
2306     /**
2307     * Universal tool change settings.
2308     * Applies to all types of extruders except where explicitly noted.
2309     */
2310 #if HAS_MULTI_EXTRUDER
2311     // Z raise distance for tool-change, as needed for some extruders
2312     #define TOOLCHANGE_ZRAISE            2 // (mm)
2313     // #define TOOLCHANGE_ZRAISE_BEFORE_RETRACT // Apply raise before swap
2314     retraction (if enabled)
2315     // #define TOOLCHANGE_NO_RETURN        // Never return to previous
2316     position on tool-change
2317     #if ENABLED(TOOLCHANGE_NO_RETURN)
2318         // #define EVENT_GCODE_AFTER_TOOLCHANGE "G12X" // Extra G-code to run
2319         after tool-change
2320     #endif
2321 /**
2322     * Extra G-code to run while executing tool-change commands. Can be used
2323     to use an additional
2324     * stepper motor (I axis, see option LINEAR_AXES in Configuration.h) to
2325     drive the tool-changer.
2326 */
2327     // #define EVENT_GCODE_TOOLCHANGE_T0 "G28 A\nG1 A0" // Extra G-code to run
2328     while executing tool-change command T0
2329     // #define EVENT_GCODE_TOOLCHANGE_T1 "G1 A10"      // Extra G-code to run

```

```

while executing tool-change command T1
2316
2317  /**
2318   * Tool Sensors detect when tools have been picked up or dropped.
2319   * Requires the pins TOOL_SENSOR1_PIN, TOOL_SENSOR2_PIN, etc.
2320   */
2321 // #define TOOL_SENSOR
2322
2323 /**
2324  * Retract and prime filament on tool-change to reduce
2325  * ooze and stringing and to get cleaner transitions.
2326  */
2327 // #define TOOLCHANGE_FILAMENT_SWAP
2328 #if ENABLED(TOOLCHANGE_FILAMENT_SWAP)
2329   // Load / Unload
2330   #define TOOLCHANGE_FS_LENGTH           12 // (mm) Load / Unload
length
2331   #define TOOLCHANGE_FS_EXTRA_RESUME_LENGTH 0 // (mm) Extra length for
better restart, fine tune by LCD/Gcode)
2332   #define TOOLCHANGE_FS_RETRACT_SPEED    (50*60) // (mm/min) (Unloading)
2333   #define TOOLCHANGE_FS_UNRETRACT_SPEED (25*60) // (mm/min) (On
SINGLEN0ZZLE or Bowden loading must be slowed down)
2334
2335   // Longer prime to clean out a SINGLEN0ZZLE
2336   #define TOOLCHANGE_FS_EXTRA_PRIME      0 // (mm) Extra priming
length
2337   #define TOOLCHANGE_FS_PRIME_SPEED     (4.6*60) // (mm/min) Extra priming
feedrate
2338   #define TOOLCHANGE_FS_WIPE_RETRACT     0 // (mm/min) Retract before
cooling for less stringing, better wipe, etc.
2339
2340   // Cool after prime to reduce stringing
2341   #define TOOLCHANGE_FS_FAN             -1 // Fan index or -1 to skip
2342   #define TOOLCHANGE_FS_FAN_SPEED       255 // 0-255
2343   #define TOOLCHANGE_FS_FAN_TIME        10 // (seconds)
2344
2345   // Swap uninitialized extruder with TOOLCHANGE_FS_PRIME_SPEED for all
lengths (recover + prime)
2346   // (May break filament if not retracted beforehand.)
2347   // #define TOOLCHANGE_FS_INIT_BEFORE_SWAP
2348
2349   // Prime on the first T0 (If other, TOOLCHANGE_FS_INIT_BEFORE_SWAP
applied)
2350   // Enable it (M217 V[0/1]) before printing, to avoid unwanted priming on
host connect
2351   // #define TOOLCHANGE_FS_PRIME_FIRST_USED
2352
2353 /**
2354  * Tool Change Migration
2355  * This feature provides G-code and LCD options to switch tools mid-
print.
2356  * All applicable tool properties are migrated so the print can
continue.
2357  * Tools must be closely matching and other restrictions may apply.
2358  * Useful to:
2359  *   - Change filament color without interruption
2360  *   - Switch spools automatically on filament runout
2361  *   - Switch to a different nozzle on an extruder jam
2362 */

```

```

2362
2363 #define TOOLCHANGE_MIGRATION_FEATURE
2364
2365 #endif
2366
2367 /**
2368 * Position to park head during tool change.
2369 * Doesn't apply to SWITCHING_TOOLHEAD, DUAL_X_CARRIAGE, or
2370 PARKING_EXTRUDER
2371 */
2372 // #define TOOLCHANGE_PARK
2373 #if ENABLED(TOOLCHANGE_PARK)
2374 #define TOOLCHANGE_PARK_XY { X_MIN_POS + 10, Y_MIN_POS + 10 }
2375 #define TOOLCHANGE_PARK_XY_FEEDRATE 6000 // (mm/min)
2376 // #define TOOLCHANGE_PARK_X_ONLY // X axis only move
2377 // #define TOOLCHANGE_PARK_Y_ONLY // Y axis only move
2378#endif
2379#endif // HAS_MULTI_EXTRUDER
2380
2381 /**
2382 * Advanced Pause for Filament Change
2383 * - Adds the G-code M600 Filament Change to initiate a filament change.
2384 * - This feature is required for the default FILAMENT_RUNOUT_SCRIPT.
2385 *
2386 * Requirements:
2387 * - For Filament Change parking enable and configure NOZZLE_PARK_FEATURE.
2388 * - For user interaction enable an LCD display, HOST_PROMPT_SUPPORT, or
2389 EMERGENCY_PARSER.
2390 *
2391 * Enable PARK_HEAD_ON_PAUSE to add the G-code M125 Pause and Park.
2392 */
2393 // #define ADVANCED_PAUSE_FEATURE
2394 #if ENABLED(ADVANCED_PAUSE_FEATURE)
2395 #define PAUSE_PARK_RETRACT_FEEDRATE 60 // (mm/s) Initial retract
2396 feedrate.
2397 #define PAUSE_PARK_RETRACT_LENGTH 2 // (mm) Initial retract.
2398 done immediately, before parking the nozzle. // This short retract is
2399 #define FILAMENT_CHANGE_UNLOAD_FEEDRATE 10 // (mm/s) Unload filament
2400 feedrate. This can be pretty fast.
2401 #define FILAMENT_CHANGE_UNLOAD_ACCEL 25 // (mm/s^2) Lower
2402 acceleration may allow a faster feedrate.
2403 #define FILAMENT_CHANGE_UNLOAD_LENGTH 100 // (mm) The length of
2404 filament for a complete unload.
2405 length of the tube and nozzle. // For Bowden, the full
2406 // For direct drive, the
2407 full length of the nozzle. // Set to 0 for manual
2408 unloading.
2409 #define FILAMENT_CHANGE_SLOW_LOAD_FEEDRATE 6 // (mm/s) Slow move when
2410 starting load.
2411 #define FILAMENT_CHANGE_SLOW_LOAD_LENGTH 0 // (mm) Slow length, to
2412 allow time to insert material. // 0 to disable start
2413 loading and skip to fast load only
2414 #define FILAMENT_CHANGE_FAST_LOAD_FEEDRATE 6 // (mm/s) Load filament
2415 feedrate. This can be pretty fast.
2416 #define FILAMENT_CHANGE_FAST_LOAD_ACCEL 25 // (mm/s^2) Lower

```

```

acceleration may allow a faster feedrate.
2407 #define FILAMENT_CHANGE_FAST_LOAD_LENGTH      0 // (mm) Load length of
filament, from extruder gear to nozzle.                                // For Bowden, the full
2408 length of the tube and nozzle.                                         // For direct drive, the
2409 full length of the nozzle.                                            // Purge continuously up
2410 // #define ADVANCED_PAUSE_CONTINUOUS_PURGE      3 // (mm/s) Extrude feedrate
to the purge length until interrupted.                                     // after loading). Should be slower than load feedrate.
2411 #define ADVANCED_PAUSE_PURGE_FEEDRATE          50 // (mm) Length to extrude
(after loading). Should be slower than load feedrate.                      // after loading.
2412 #define ADVANCED_PAUSE_PURGE_LENGTH            // Set to 0 for manual
2413 after loading.                                                       // extrusion.
2414 extruded repeatedly from the Filament Change menu
2415 consistent, and to purge old filament.                                // Filament can be
2416 #define ADVANCED_PAUSE_RESUME_PRIME           0 // (mm) Extra distance to
prime nozzle after returning from park.                                    // until extrusion is
2417 // #define ADVANCED_PAUSE_FANS_PAUSE          // Turn off print-cooling
fans while the machine is paused.                                         // Filament Unload does a
2418
2419 Retract, Delay, and Purge first:
2420 #define FILAMENT_UNLOAD_PURGE_RETRACT         13 // (mm) Unload initial
retract length.                                                       // filament to cool after retract.
2421 #define FILAMENT_UNLOAD_PURGE_DELAY           5000 // (ms) Delay for the
filament to cool after retract.                                         // done, then this length is purged.
2422 #define FILAMENT_UNLOAD_PURGE_LENGTH          8 // (mm) An unretract is
done, then this length is purged.                                         // #define FILAMENT_UNLOAD_PURGE_FEEDRATE
2423 #define FILAMENT_UNLOAD_PURGE_FEEDRATE        25 // (mm/s) feedrate to
purge before unload
2424
2425 #define PAUSE_PARK_NOZZLE_TIMEOUT             45 // (seconds) Time limit
before the nozzle is turned off for safety.                               // Number of alert beeps
2426 #define FILAMENT_CHANGE_ALERT_BEEPS          10 // play when a response is needed.
2427 #define PAUSE_PARK_NO_STEPPER_TIMEOUT         // Enable for XYZ steppers
to stay powered on during filament change.                               // Automatically continue
2428 // #define FILAMENT_CHANGE_RESUME_ON_INSERT   / load filament when runout sensor is triggered again.
2429 // #define PAUSE_REHEAT_FAST_RESUME          // Reduce number of waits
by not prompting again post-timeout before continuing.
2430
2431 // #define PARK_HEAD_ON_PAUSE                // Park the nozzle during
pause and filament change.                                              // If needed, home before
2432 // #define HOME_BEFORE_FILAMENT_CHANGE        parking for filament change
2433
2434 // #define FILAMENT_LOAD_UNLOAD_GCODES        // Add M701/M702
Load/Unload G-codes, plus Load/Unload in the LCD Prepare menu.
2435 // #define FILAMENT_UNLOAD_ALL_EXTRUDERS      // Allow M702 to unload
all extruders above a minimum target temp (as set by M302)
2436 #endif
2437
2438 // @section tmc
2439

```

```
2439
2440 /**
2441 * TMC26X Stepper Driver options
2442 *
2443 * The TMC26XStepper library is required for this stepper driver.
2444 * https://github.com/trinamic/TMC26XStepper
2445 */
2446 #if HAS_DRIVER(TMC26X)
2447
2448 #if AXIS_DRIVER_TYPE_X(TMC26X)
2449     #define X_MAX_CURRENT    1000 // (mA)
2450     #define X_SENSE_RESISTOR  91   // (mOhms)
2451     #define X_MICROSTEPS      16   // Number of microsteps
2452 #endif
2453
2454 #if AXIS_DRIVER_TYPE_X2(TMC26X)
2455     #define X2_MAX_CURRENT   1000
2456     #define X2_SENSE_RESISTOR 91
2457     #define X2_MICROSTEPS     X_MICROSTEPS
2458 #endif
2459
2460 #if AXIS_DRIVER_TYPE_Y(TMC26X)
2461     #define Y_MAX_CURRENT    1000
2462     #define Y_SENSE_RESISTOR  91
2463     #define Y_MICROSTEPS      16
2464 #endif
2465
2466 #if AXIS_DRIVER_TYPE_Y2(TMC26X)
2467     #define Y2_MAX_CURRENT   1000
2468     #define Y2_SENSE_RESISTOR 91
2469     #define Y2_MICROSTEPS     Y_MICROSTEPS
2470 #endif
2471
2472 #if AXIS_DRIVER_TYPE_Z(TMC26X)
2473     #define Z_MAX_CURRENT    1000
2474     #define Z_SENSE_RESISTOR  91
2475     #define Z_MICROSTEPS      16
2476 #endif
2477
2478 #if AXIS_DRIVER_TYPE_Z2(TMC26X)
2479     #define Z2_MAX_CURRENT   1000
2480     #define Z2_SENSE_RESISTOR 91
2481     #define Z2_MICROSTEPS     Z_MICROSTEPS
2482 #endif
2483
2484 #if AXIS_DRIVER_TYPE_Z3(TMC26X)
2485     #define Z3_MAX_CURRENT   1000
2486     #define Z3_SENSE_RESISTOR 91
2487     #define Z3_MICROSTEPS     Z_MICROSTEPS
2488 #endif
2489
2490 #if AXIS_DRIVER_TYPE_Z4(TMC26X)
2491     #define Z4_MAX_CURRENT   1000
2492     #define Z4_SENSE_RESISTOR 91
2493     #define Z4_MICROSTEPS     Z_MICROSTEPS
2494 #endif
2495
2496 #if AXIS_DRIVER_TYPE_I(TMC26X)
2497     #define I_MAX_CURRENT    1000
```

```
2498 #define I_SENSE_RESISTOR 91
2499 #define I_MICROSTEPS 16
2500 #endif
2501
2502 #if AXIS_DRIVER_TYPE_J(TMC26X)
2503     #define J_MAX_CURRENT 1000
2504     #define J_SENSE_RESISTOR 91
2505     #define J_MICROSTEPS 16
2506 #endif
2507
2508 #if AXIS_DRIVER_TYPE_K(TMC26X)
2509     #define K_MAX_CURRENT 1000
2510     #define K_SENSE_RESISTOR 91
2511     #define K_MICROSTEPS 16
2512 #endif
2513
2514 #if AXIS_DRIVER_TYPE_E0(TMC26X)
2515     #define E0_MAX_CURRENT 1000
2516     #define E0_SENSE_RESISTOR 91
2517     #define E0_MICROSTEPS 16
2518 #endif
2519
2520 #if AXIS_DRIVER_TYPE_E1(TMC26X)
2521     #define E1_MAX_CURRENT 1000
2522     #define E1_SENSE_RESISTOR 91
2523     #define E1_MICROSTEPS E0_MICROSTEPS
2524 #endif
2525
2526 #if AXIS_DRIVER_TYPE_E2(TMC26X)
2527     #define E2_MAX_CURRENT 1000
2528     #define E2_SENSE_RESISTOR 91
2529     #define E2_MICROSTEPS E0_MICROSTEPS
2530 #endif
2531
2532 #if AXIS_DRIVER_TYPE_E3(TMC26X)
2533     #define E3_MAX_CURRENT 1000
2534     #define E3_SENSE_RESISTOR 91
2535     #define E3_MICROSTEPS E0_MICROSTEPS
2536 #endif
2537
2538 #if AXIS_DRIVER_TYPE_E4(TMC26X)
2539     #define E4_MAX_CURRENT 1000
2540     #define E4_SENSE_RESISTOR 91
2541     #define E4_MICROSTEPS E0_MICROSTEPS
2542 #endif
2543
2544 #if AXIS_DRIVER_TYPE_E5(TMC26X)
2545     #define E5_MAX_CURRENT 1000
2546     #define E5_SENSE_RESISTOR 91
2547     #define E5_MICROSTEPS E0_MICROSTEPS
2548 #endif
2549
2550 #if AXIS_DRIVER_TYPE_E6(TMC26X)
2551     #define E6_MAX_CURRENT 1000
2552     #define E6_SENSE_RESISTOR 91
2553     #define E6_MICROSTEPS E0_MICROSTEPS
2554 #endif
2555
2556 #include "AVTC_DRIVER.h"
2557
```

```

2550 #define HAS_TMC_DRIVER_TYPE_E / (TMC26X)
2551     #define E7_MAX_CURRENT    1000
2552     #define E7_SENSE_RESISTOR 91
2553     #define E7_MICROSTEPS      E0_MICROSTEPS
2554 #endif
2555
2556#endif // TMC26X
2557
2558// @section tmc_smart
2559
2560/***
2561 * To use TMC2130, TMC2160, TMC2660, TMC5130, TMC5160 stepper drivers in SPI
2562 mode
2563 * connect your SPI pins to the hardware SPI interface on your board and
2564 define
2565 * the required CS pins in your `pins_MYBOARD.h` file. (e.g., RAMPS 1.4 uses
2566 AUX3
2567 * pins `X_CS_PIN 53`, `Y_CS_PIN 49`, etc.).
2568 * You may also use software SPI if you wish to use general purpose IO pins.
2569 *
2570 * To use TMC2208 stepper UART-configurable stepper drivers connect
2571 *_SERIAL_TX_PIN
2572 * to the driver side PDN_UART pin with a 1K resistor.
2573 * To use the reading capabilities, also connect *_SERIAL_RX_PIN to PDN_UART
2574 without
2575 * a resistor.
2576 * The drivers can also be used with hardware serial.
2577 *
2578 * TMCStepper library is required to use TMC stepper drivers.
2579 * https://github.com/teemuatlut/TMCStepper
2580 */
2581
2582#if HAS_TRINAMIC_CONFIG
2583
2584#define HOLD_MULTIPLIER    0.5 // Scales down the holding current from
2585run current
2586
2587/***
2588 * Interpolate microsteps to 256
2589 * Override for each driver with <driver>_INTERPOLATE settings below
2590 */
2591#define INTERPOLATE        true
2592
2593#if AXIS_IS_TMC(X)
2594    #define X_CURRENT        800 // (mA) RMS current. Multiply by
25951.414 for peak current.
2596    #define X_CURRENT_HOME   X_CURRENT // (mA) RMS current for sensorless
2597homing
2598    #define X_MICROSTEPS     16 // 0..256
2599    #define X_RSENSE          0.11
2600    #define X_CHAIN_POS       -1 // -1..0: Not chained. 1: MCU MOSI
2601connected. 2: Next in chain, ...
2602    // #define X_INTERPOLATE true // Enable to override 'INTERPOLATE'
2603for the X axis
2604#endif
2605
2606#if AXIS_IS_TMC(X2)
2607    #define X2_CURRENT        800
2608    #define X2_CURRENT_HOME   X2_CURRENT
2609    #define X2_MICROSTEPS     X_MICROSTEPS

```

```
2605 #define X2_RSENSE      0.11
2606 #define X2_CHAIN_POS    -1
2607 //#define X2_INTERPOLATE true
2608#endif
2609
2610#if AXIS_IS_TMC(Y)
2611#define Y_CURRENT      800
2612#define Y_CURRENT_HOME Y_CURRENT
2613#define Y_MICROSTEPS   16
2614#define Y_RSENSE        0.11
2615#define Y_CHAIN_POS     -1
2616//#define Y_INTERPOLATE true
2617#endif
2618
2619#if AXIS_IS_TMC(Y2)
2620#define Y2_CURRENT      800
2621#define Y2_CURRENT_HOME Y2_CURRENT
2622#define Y2_MICROSTEPS   Y_MICROSTEPS
2623#define Y2_RSENSE        0.11
2624#define Y2_CHAIN_POS     -1
2625//#define Y2_INTERPOLATE true
2626#endif
2627
2628#if AXIS_IS_TMC(Z)
2629#define Z_CURRENT      800
2630#define Z_CURRENT_HOME Z_CURRENT
2631#define Z_MICROSTEPS   16
2632#define Z_RSENSE        0.11
2633#define Z_CHAIN_POS     -1
2634//#define Z_INTERPOLATE true
2635#endif
2636
2637#if AXIS_IS_TMC(Z2)
2638#define Z2_CURRENT      800
2639#define Z2_CURRENT_HOME Z2_CURRENT
2640#define Z2_MICROSTEPS   Z_MICROSTEPS
2641#define Z2_RSENSE        0.11
2642#define Z2_CHAIN_POS     -1
2643//#define Z2_INTERPOLATE true
2644#endif
2645
2646#if AXIS_IS_TMC(Z3)
2647#define Z3_CURRENT      800
2648#define Z3_CURRENT_HOME Z3_CURRENT
2649#define Z3_MICROSTEPS   Z_MICROSTEPS
2650#define Z3_RSENSE        0.11
2651#define Z3_CHAIN_POS     -1
2652//#define Z3_INTERPOLATE true
2653#endif
2654
2655#if AXIS_IS_TMC(Z4)
2656#define Z4_CURRENT      800
2657#define Z4_CURRENT_HOME Z4_CURRENT
2658#define Z4_MICROSTEPS   Z_MICROSTEPS
2659#define Z4_RSENSE        0.11
2660#define Z4_CHAIN_POS     -1
2661//#define Z4_INTERPOLATE true
2662#endif
```

```
2663  
2664 #if AXIS_IS_TMC(I)  
2665     #define I_CURRENT      800  
2666     #define I_CURRENT_HOME I_CURRENT  
2667     #define I_MICROSTEPS   16  
2668     #define I_RSENSE        0.11  
2669     #define I_CHAIN_POS     -1  
2670     //#define I_INTERPOLATE true  
2671 #endif  
2672  
2673 #if AXIS_IS_TMC(J)  
2674     #define J_CURRENT      800  
2675     #define J_CURRENT_HOME J_CURRENT  
2676     #define J_MICROSTEPS   16  
2677     #define J_RSENSE        0.11  
2678     #define J_CHAIN_POS     -1  
2679     //#define J_INTERPOLATE true  
2680 #endif  
2681  
2682 #if AXIS_IS_TMC(K)  
2683     #define K_CURRENT      800  
2684     #define K_CURRENT_HOME K_CURRENT  
2685     #define K_MICROSTEPS   16  
2686     #define K_RSENSE        0.11  
2687     #define K_CHAIN_POS     -1  
2688     //#define K_INTERPOLATE true  
2689 #endif  
2690  
2691 #if AXIS_IS_TMC(E0)  
2692     #define E0_CURRENT      800  
2693     #define E0_MICROSTEPS   16  
2694     #define E0_RSENSE        0.11  
2695     #define E0_CHAIN_POS     -1  
2696     //#define E0_INTERPOLATE true  
2697 #endif  
2698  
2699 #if AXIS_IS_TMC(E1)  
2700     #define E1_CURRENT      800  
2701     #define E1_MICROSTEPS   E0_MICROSTEPS  
2702     #define E1_RSENSE        0.11  
2703     #define E1_CHAIN_POS     -1  
2704     //#define E1_INTERPOLATE true  
2705 #endif  
2706  
2707 #if AXIS_IS_TMC(E2)  
2708     #define E2_CURRENT      800  
2709     #define E2_MICROSTEPS   E0_MICROSTEPS  
2710     #define E2_RSENSE        0.11  
2711     #define E2_CHAIN_POS     -1  
2712     //#define E2_INTERPOLATE true  
2713 #endif  
2714  
2715 #if AXIS_IS_TMC(E3)  
2716     #define E3_CURRENT      800  
2717     #define E3_MICROSTEPS   E0_MICROSTEPS  
2718     #define E3_RSENSE        0.11  
2719     #define E3_CHAIN_POS     -1  
2720     //#define E3_INTERPOLATE true  
2721 #endif
```

```
2722  
2723 #if AXIS_IS_TMC(E4)  
2724     #define E4_CURRENT      800  
2725     #define E4_MICROSTEPS   E0_MICROSTEPS  
2726     #define E4_RSENSE        0.11  
2727     #define E4_CHAIN_POS     -1  
2728     //#define E4_INTERPOLATE true  
2729 #endif  
2730  
2731 #if AXIS_IS_TMC(E5)  
2732     #define E5_CURRENT      800  
2733     #define E5_MICROSTEPS   E0_MICROSTEPS  
2734     #define E5_RSENSE        0.11  
2735     #define E5_CHAIN_POS     -1  
2736     //#define E5_INTERPOLATE true  
2737 #endif  
2738  
2739 #if AXIS_IS_TMC(E6)  
2740     #define E6_CURRENT      800  
2741     #define E6_MICROSTEPS   E0_MICROSTEPS  
2742     #define E6_RSENSE        0.11  
2743     #define E6_CHAIN_POS     -1  
2744     //#define E6_INTERPOLATE true  
2745 #endif  
2746  
2747 #if AXIS_IS_TMC(E7)  
2748     #define E7_CURRENT      800  
2749     #define E7_MICROSTEPS   E0_MICROSTEPS  
2750     #define E7_RSENSE        0.11  
2751     #define E7_CHAIN_POS     -1  
2752     //#define E7_INTERPOLATE true  
2753 #endif  
2754  
2755 /**  
2756  * Override default SPI pins for TMC2130, TMC2160, TMC2660, TMC5130 and  
TMC5160 drivers here.  
2757  * The default pins can be found in your board's pins file.  
2758  */  
2759 //##define X_CS_PIN          -1  
2760 //##define Y_CS_PIN          -1  
2761 //##define Z_CS_PIN          -1  
2762 //##define X2_CS_PIN         -1  
2763 //##define Y2_CS_PIN         -1  
2764 //##define Z2_CS_PIN         -1  
2765 //##define Z3_CS_PIN         -1  
2766 //##define Z4_CS_PIN         -1  
2767 //##define I_CS_PIN          -1  
2768 //##define J_CS_PIN          -1  
2769 //##define K_CS_PIN          -1  
2770 //##define E0_CS_PIN         -1  
2771 //##define E1_CS_PIN         -1  
2772 //##define E2_CS_PIN         -1  
2773 //##define E3_CS_PIN         -1  
2774 //##define E4_CS_PIN         -1  
2775 //##define E5_CS_PIN         -1  
2776 //##define E6_CS_PIN         -1  
2777 //##define E7_CS_PIN         -1  
2778 ,
```

```

2781 /**
2782  * Software option for SPI driven drivers (TMC2130, TMC2160, TMC2660,
2783 TMC5130 and TMC5160).
2784  * The default SW SPI pins are defined the respective pins files,
2785  * but you can override or define them here.
2786 */
2787 //#define TMC_USE_SW_SPI
2788 //#define TMC_SW_MOSI      -1
2789 //#define TMC_SW_MISO      -1
2790 //#define TMC_SW_SCK       -1
2791 /**
2792  * Four TMC2209 drivers can use the same HW/SW serial port with hardware
2793 configured addresses.
2794  * Set the address using jumpers on pins MS1 and MS2.
2795  * Address | MS1   | MS2
2796  *          0 | LOW   | LOW
2797  *          1 | HIGH  | LOW
2798  *          2 | LOW   | HIGH
2799  *          3 | HIGH  | HIGH
2800 */
2801 //#define X_SLAVE_ADDRESS 0
2802 //#define Y_SLAVE_ADDRESS 0
2803 //#define Z_SLAVE_ADDRESS 0
2804 //#define X2_SLAVE_ADDRESS 0
2805 //#define Y2_SLAVE_ADDRESS 0
2806 //#define Z2_SLAVE_ADDRESS 0
2807 //#define Z3_SLAVE_ADDRESS 0
2808 //#define Z4_SLAVE_ADDRESS 0
2809 //#define I_SLAVE_ADDRESS 0
2810 //#define J_SLAVE_ADDRESS 0
2811 //#define K_SLAVE_ADDRESS 0
2812 //#define E0_SLAVE_ADDRESS 0
2813 //#define E1_SLAVE_ADDRESS 0
2814 //#define E2_SLAVE_ADDRESS 0
2815 //#define E3_SLAVE_ADDRESS 0
2816 //#define E4_SLAVE_ADDRESS 0
2817 //#define E5_SLAVE_ADDRESS 0
2818 //#define E6_SLAVE_ADDRESS 0
2819 //#define E7_SLAVE_ADDRESS 0
2820
2821 /**
2822  * Software enable
2823  *
2824  * Use for drivers that do not use a dedicated enable pin, but rather
2825 handle the same
2826  * function through a communication line such as SPI or UART.
2827 */
2828 //#define SOFTWARE_DRIVER_ENABLE
2829
2830 /**
2831  * TMC2130, TMC2160, TMC2208, TMC2209, TMC5130 and TMC5160 only
2832  * Use Trinamic's ultra quiet stepping mode.
2833  * When disabled, Marlin will use spreadCycle stepping mode.
2834 */
#define STEALTHCHOP XY

```

```

2835 #define STEALTHCHOP_Z
2836 #define STEALTHCHOP_I
2837 #define STEALTHCHOP_J
2838 #define STEALTHCHOP_K
2839 #define STEALTHCHOP_E
2840
2841 /**
2842 * Optimize spreadCycle chopper parameters by using predefined parameter
2843 sets
2844 * or with the help of an example included in the library.
2845 * Provided parameter sets are
2846 * CHOPPER_DEFAULT_12V
2847 * CHOPPER_DEFAULT_19V
2848 * CHOPPER_DEFAULT_24V
2849 * CHOPPER_DEFAULT_36V
2850 * CHOPPER_09STEP_24V // 0.9 degree steppers (24V)
2851 * CHOPPER_PRUSAMK3_24V // Imported parameters from the official Průša
firmware for MK3 (24V)
2852 * CHOPPER_MARLIN_119 // Old defaults from Marlin v1.1.9
2853 *
2854 * Define your own with:
2855 * { <off_time[1..15]>, <hysteresis_end[-3..12]>, hysteresis_start[1..8] }
2856 */
2857 #define CHOPPER_TIMING CHOPPER_DEFAULT_12V // All axes (override
below)
2858 // #define CHOPPER_TIMING_X CHOPPER_TIMING // For X Axes (override
below)
2859 // #define CHOPPER_TIMING_X2 CHOPPER_TIMING_X // For Y Axes (override
below)
2860 // #define CHOPPER_TIMING_Y CHOPPER_TIMING_Y // For Z Axes (override
below)
2861 // #define CHOPPER_TIMING_Y2 CHOPPER_TIMING_Y
2862 // #define CHOPPER_TIMING_Z CHOPPER_TIMING_Z
2863 // #define CHOPPER_TIMING_Z2 CHOPPER_TIMING_Z
2864 // #define CHOPPER_TIMING_Z3 CHOPPER_TIMING_Z
2865 // #define CHOPPER_TIMING_Z4 CHOPPER_TIMING_Z
2866 // #define CHOPPER_TIMING_E CHOPPER_TIMING_E
2867 // #define CHOPPER_TIMING_E1 CHOPPER_TIMING_E
2868 // #define CHOPPER_TIMING_E2 CHOPPER_TIMING_E
2869 // #define CHOPPER_TIMING_E3 CHOPPER_TIMING_E
2870 // #define CHOPPER_TIMING_E4 CHOPPER_TIMING_E
2871 // #define CHOPPER_TIMING_E5 CHOPPER_TIMING_E
2872 // #define CHOPPER_TIMING_E6 CHOPPER_TIMING_E
2873 // #define CHOPPER_TIMING_E7 CHOPPER_TIMING_E
2874 /**
2875 * Monitor Trinamic drivers
2876 * for error conditions like overtemperature and short to ground.
2877 * To manage over-temp Marlin can decrease the driver current until the
error condition clears.
2878 * Other detected conditions can be used to stop the current print.
2879 * Relevant G-codes:
2880 * M906 – Set or get motor current in millamps using axis codes X, Y, Z,
E. Report values if no axis codes given.
2881 * M911 – Report stepper driver overtemperature pre-warn condition.
2882 * M912 – Clear stepper driver overtemperature pre-warn condition flag.
2883 * M122 – Report driver parameters (Requires TMC_DEBUG)

```

```

2884 */
2885 // #define MONITOR_DRIVER_STATUS
2886
2887 #if ENABLED(MONITOR_DRIVER_STATUS)
2888     #define CURRENT_STEP_DOWN      50 // [mA]
2889     #define REPORT_CURRENT_CHANGE
2890     #define STOP_ON_ERROR
2891 #endif
2892
2893 /**
2894 * TMC2130, TMC2160, TMC2208, TMC2209, TMC5130 and TMC5160 only
2895 * The driver will switch to spreadCycle when stepper speed is over
2896 HYBRID_THRESHOLD.
2897 * This mode allows for faster movements at the expense of higher noise
2898 levels.
2899 * STEALTHCHOP_(XY|Z|E) must be enabled to use HYBRID_THRESHOLD.
2900 * M913 X/Y/Z/E to live tune the setting
2901 */
2902 // #define HYBRID_THRESHOLD
2903
2904 #define X_HYBRID_THRESHOLD      100 // [mm/s]
2905 #define X2_HYBRID_THRESHOLD    100
2906 #define Y_HYBRID_THRESHOLD      100
2907 #define Y2_HYBRID_THRESHOLD    100
2908 #define Z_HYBRID_THRESHOLD        3
2909 #define Z2_HYBRID_THRESHOLD      3
2910 #define Z3_HYBRID_THRESHOLD      3
2911 #define Z4_HYBRID_THRESHOLD      3
2912 #define I_HYBRID_THRESHOLD        3
2913 #define J_HYBRID_THRESHOLD        3
2914 #define K_HYBRID_THRESHOLD        3
2915 #define E0_HYBRID_THRESHOLD      30
2916 #define E1_HYBRID_THRESHOLD      30
2917 #define E2_HYBRID_THRESHOLD      30
2918 #define E3_HYBRID_THRESHOLD      30
2919 #define E4_HYBRID_THRESHOLD      30
2920 #define E5_HYBRID_THRESHOLD      30
2921 #define E6_HYBRID_THRESHOLD      30
2922 #define E7_HYBRID_THRESHOLD      30
2923
2924 /**
2925 * Use StallGuard to home / probe X, Y, Z.
2926 *
2927 * TMC2130, TMC2160, TMC2209, TMC2660, TMC5130, and TMC5160 only
2928 * Connect the stepper driver's DIAG1 pin to the X/Y endstop pin.
2929 * X, Y, and Z homing will always be done in spreadCycle mode.
2930 *
2931 * X/Y/Z_STALL_SENSITIVITY is the default stall threshold.
2932 * Use M914 X Y Z to set the stall threshold at runtime:
2933 *
2934 * Sensitivity   TMC2209   Others
2935 * HIGHEST       255       -64   (Too sensitive => False positive)
2936 * LOWEST        0         63   (Too insensitive => No trigger)
2937 *
2938 * It is recommended to set HOMING_BUMP_MM to { 0, 0, 0 }.
2939 *
2940 * SPI_ENDSTOPS *** Beta feature! *** TMC2130/TMC5160 Only ***
2941 * Poll the driver through SPI to determine load when homing.
2942 * Removes the need for a wire from DTAG1 to an endstop pin.

```

```

2941 *
2942 * IMPROVE_HOMING_RELIABILITY tunes acceleration and jerk when
2943 * homing and adds a guard period for endstop triggering.
2944 *
2945 * Comment *_STALL_SENSITIVITY to disable sensorless homing for that axis.
2946 */
2947 // #define SENSORLESS_HOMING // StallGuard capable drivers only
2948
2949 #if EITHER(SENSORLESS_HOMING, SENSORLESS_PROBING)
2950     // TMC2209: 0...255. TMC2130: -64...63
2951     #define X_STALL_SENSITIVITY 8
2952     #define X2_STALL_SENSITIVITY X_STALL_SENSITIVITY
2953     #define Y_STALL_SENSITIVITY 8
2954     #define Y2_STALL_SENSITIVITY Y_STALL_SENSITIVITY
2955     // #define Z_STALL_SENSITIVITY 8
2956     // #define Z2_STALL_SENSITIVITY Z_STALL_SENSITIVITY
2957     // #define Z3_STALL_SENSITIVITY Z_STALL_SENSITIVITY
2958     // #define Z4_STALL_SENSITIVITY Z_STALL_SENSITIVITY
2959     // #define I_STALL_SENSITIVITY 8
2960     // #define J_STALL_SENSITIVITY 8
2961     // #define K_STALL_SENSITIVITY 8
2962     // #define SPI_ENDSTOPS           // TMC2130 only
2963     // #define IMPROVE_HOMING_RELIABILITY
2964 #endif
2965
2966 /**
2967 * TMC Homing stepper phase.
2968 *
2969 * Improve homing repeatability by homing to stepper coil's nearest
2970 * absolute
2971 * phase position. Trinamic drivers use a stepper phase table with 1024
2972 * values
2973 * spanning 4 full steps with 256 positions each (ergo, 1024 positions).
2974 * Full step positions (128, 384, 640, 896) have the highest holding
2975 * torque.
2976 *
2977 * Values from 0..1023, -1 to disable homing phase for that axis.
2978 */
2979 // #define TMC_HOME_PHASE { 896, 896, 896 }
2980
2981 /**
2982 * Beta feature!
2983 * Create a 50/50 square wave step pulse optimal for stepper drivers.
2984 */
2985 // #define SQUARE_WAVE_STEPPING
2986
2987 /**
2988 * Enable M122 debugging command for TMC stepper drivers.
2989 * M122 S0/1 will enable continuous reporting.
2990 */
2991 // #define TMC_DEBUG
2992
2993 /**
2994 * You can set your own advanced settings by filling in predefined
2995 * functions.
2996 * A list of available functions can be found on the library github page
2997 * https://github.com/teemuatlut/TMCStepper
2998 */

```

```

2995 * Example:
2996 * #define TMC_ADV() { \
2997 *   stepperX.diag0_otpw(1); \
2998 *   stepperY.interp(0); \
2999 * }
3000 */
3001 #define TMC_ADV() { }
3002
3003#endif // HAS_TRINAMIC_CONFIG
3004
3005// @section L64XX
3006
3007/**
3008 * L64XX Stepper Driver options
3009 *
3010 * Arduino-L6470 library (0.8.0 or higher) is required.
3011 * https://github.com/ameyer/Arduino-L6470
3012 *
3013 * Requires the following to be defined in your pins_YOUR_BOARD file
3014 *     L6470_CHAIN_SCK_PIN
3015 *     L6470_CHAIN_MISO_PIN
3016 *     L6470_CHAIN_MOSI_PIN
3017 *     L6470_CHAIN_SS_PIN
3018 *     ENABLE_RESET_L64XX_CHIPS(Q) where Q is 1 to enable and 0 to reset
3019 */
3020
3021#if HAS_L64XX
3022
3023    // #define L6470_CHITCHAT           // Display additional status info
3024
3025    #if AXIS_IS_L64XX(X)
3026        #define X_MICROSTEPS      128 // Number of microsteps (VALID: 1, 2, 4,
3027        8, 16, 32, 128) - L6474 max is 16
3028        #define X_OVERCURRENT     2000 // (mA) Current where the driver detects
3029        an over current
3030
3031        // L6470 & L6474 - VALID: 375 x (1 -
3032        16) - 6A max - rounds down
3033
3034        // POWERSTEP01: VALID: 1000 x (1 - 32)
3035        - 32A max - rounds down
3036        #define X_STALLCURRENT    1500 // (mA) Current where the driver detects
3037        a stall (VALID: 31.25 * (1-128) - 4A max - rounds down)
3038
3039        // L6470 & L6474 - VALID: 31.25 * (1-
3040        128) - 4A max - rounds down
3041
3042        // POWERSTEP01: VALID: 200 x (1 - 32)
3043        - 6.4A max - rounds down
3044
3045        // L6474 - STALLCURRENT setting is
3046        used to set the nominal (TVAL) current
3047        #define X_MAX_VOLTAGE     127 // 0-255, Maximum effective voltage seen
3048        by stepper - not used by L6474
3049        #define X_CHAIN_POS        -1 // Position in SPI chain, 0=Not in
3050        chain, 1=Nearest MOSI
3051        #define X_SLEW_RATE        1 // 0-3, Slew 0 is slowest, 3 is fastest
3052    #endif
3053
3054    #if AXIS_IS_L64XX(X2)
3055        #define X2_MICROSTEPS     X_MICROSTEPS
3056        #define X2_OVERCURRENT    2000
3057        #define X2_STALLCURRENT   1500
3058        #define X2_MAX_VOLTAGE    127
3059    #endif

```

```

3044 #define X2_CHAIN_POS          -1
3045 #define X2_SLEW_RATE          1
3046 #endif
3047
3048 #if AXIS_IS_L64XX(Y)
3049     #define Y_MICROSTEPS      128
3050     #define Y_OVERCURRENT    2000
3051     #define Y_STALLCURRENT   1500
3052     #define Y_MAX_VOLTAGE    127
3053     #define Y_CHAIN_POS        -1
3054     #define Y_SLEW_RATE         1
3055 #endif
3056
3057 #if AXIS_IS_L64XX(Y2)
3058     #define Y2_MICROSTEPS     Y_MICROSTEPS
3059     #define Y2_OVERCURRENT    2000
3060     #define Y2_STALLCURRENT   1500
3061     #define Y2_MAX_VOLTAGE    127
3062     #define Y2_CHAIN_POS        -1
3063     #define Y2_SLEW_RATE         1
3064 #endif
3065
3066 #if AXIS_IS_L64XX(Z)
3067     #define Z_MICROSTEPS      128
3068     #define Z_OVERCURRENT    2000
3069     #define Z_STALLCURRENT   1500
3070     #define Z_MAX_VOLTAGE    127
3071     #define Z_CHAIN_POS        -1
3072     #define Z_SLEW_RATE         1
3073 #endif
3074
3075 #if AXIS_IS_L64XX(Z2)
3076     #define Z2_MICROSTEPS     Z_MICROSTEPS
3077     #define Z2_OVERCURRENT    2000
3078     #define Z2_STALLCURRENT   1500
3079     #define Z2_MAX_VOLTAGE    127
3080     #define Z2_CHAIN_POS        -1
3081     #define Z2_SLEW_RATE         1
3082 #endif
3083
3084 #if AXIS_IS_L64XX(Z3)
3085     #define Z3_MICROSTEPS     Z_MICROSTEPS
3086     #define Z3_OVERCURRENT    2000
3087     #define Z3_STALLCURRENT   1500
3088     #define Z3_MAX_VOLTAGE    127
3089     #define Z3_CHAIN_POS        -1
3090     #define Z3_SLEW_RATE         1
3091 #endif
3092
3093 #if AXIS_IS_L64XX(Z4)
3094     #define Z4_MICROSTEPS     Z_MICROSTEPS
3095     #define Z4_OVERCURRENT    2000
3096     #define Z4_STALLCURRENT   1500
3097     #define Z4_MAX_VOLTAGE    127
3098     #define Z4_CHAIN_POS        -1
3099     #define Z4_SLEW_RATE         1
3100 #endif
3101

```

```

3102 #if AXIS_DRIVER_TYPE_I(L6470)
3103     #define I_MICROSTEPS      128
3104     #define I_OVERCURRENT    2000
3105     #define I_STALLCURRENT   1500
3106     #define I_MAX_VOLTAGE    127
3107     #define I_CHAIN_POS       -1
3108     #define I_SLEW_RATE        1
3109 #endif
3110
3111 #if AXIS_DRIVER_TYPE_J(L6470)
3112     #define J_MICROSTEPS      128
3113     #define J_OVERCURRENT    2000
3114     #define J_STALLCURRENT   1500
3115     #define J_MAX_VOLTAGE    127
3116     #define J_CHAIN_POS       -1
3117     #define J_SLEW_RATE        1
3118 #endif
3119
3120 #if AXIS_DRIVER_TYPE_K(L6470)
3121     #define K_MICROSTEPS      128
3122     #define K_OVERCURRENT    2000
3123     #define K_STALLCURRENT   1500
3124     #define K_MAX_VOLTAGE    127
3125     #define K_CHAIN_POS       -1
3126     #define K_SLEW_RATE        1
3127 #endif
3128
3129 #if AXIS_IS_L64XX(E0)
3130     #define E0_MICROSTEPS      128
3131     #define E0_OVERCURRENT    2000
3132     #define E0_STALLCURRENT   1500
3133     #define E0_MAX_VOLTAGE    127
3134     #define E0_CHAIN_POS       -1
3135     #define E0_SLEW_RATE        1
3136 #endif
3137
3138 #if AXIS_IS_L64XX(E1)
3139     #define E1_MICROSTEPS      E0_MICROSTEPS
3140     #define E1_OVERCURRENT    2000
3141     #define E1_STALLCURRENT   1500
3142     #define E1_MAX_VOLTAGE    127
3143     #define E1_CHAIN_POS       -1
3144     #define E1_SLEW_RATE        1
3145 #endif
3146
3147 #if AXIS_IS_L64XX(E2)
3148     #define E2_MICROSTEPS      E0_MICROSTEPS
3149     #define E2_OVERCURRENT    2000
3150     #define E2_STALLCURRENT   1500
3151     #define E2_MAX_VOLTAGE    127
3152     #define E2_CHAIN_POS       -1
3153     #define E2_SLEW_RATE        1
3154 #endif
3155
3156 #if AXIS_IS_L64XX(E3)
3157     #define E3_MICROSTEPS      E0_MICROSTEPS
3158     #define E3_OVERCURRENT    2000
3159     #define E3_STALLCURRENT   1500
3160     #define E3_MAX_VOLTAGE    127

```

```

3160      #define E3_MAX_VOLTAGE          127
3161      #define E3_CHAIN_POS            -1
3162      #define E3_SLEW_RATE             1
3163 #endif
3164
3165 #if AXIS_IS_L64XX(E4)
3166     #define E4_MICROSTEPS        E0_MICROSTEPS
3167     #define E4_OVERCURRENT         2000
3168     #define E4_STALLCURRENT        1500
3169     #define E4_MAX_VOLTAGE          127
3170     #define E4_CHAIN_POS            -1
3171     #define E4_SLEW_RATE             1
3172 #endif
3173
3174 #if AXIS_IS_L64XX(E5)
3175     #define E5_MICROSTEPS        E0_MICROSTEPS
3176     #define E5_OVERCURRENT         2000
3177     #define E5_STALLCURRENT        1500
3178     #define E5_MAX_VOLTAGE          127
3179     #define E5_CHAIN_POS            -1
3180     #define E5_SLEW_RATE             1
3181 #endif
3182
3183 #if AXIS_IS_L64XX(E6)
3184     #define E6_MICROSTEPS        E0_MICROSTEPS
3185     #define E6_OVERCURRENT         2000
3186     #define E6_STALLCURRENT        1500
3187     #define E6_MAX_VOLTAGE          127
3188     #define E6_CHAIN_POS            -1
3189     #define E6_SLEW_RATE             1
3190 #endif
3191
3192 #if AXIS_IS_L64XX(E7)
3193     #define E7_MICROSTEPS        E0_MICROSTEPS
3194     #define E7_OVERCURRENT         2000
3195     #define E7_STALLCURRENT        1500
3196     #define E7_MAX_VOLTAGE          127
3197     #define E7_CHAIN_POS            -1
3198     #define E7_SLEW_RATE             1
3199 #endif
3200
3201 /**
3202  * Monitor L6470 drivers for error conditions like over temperature and
3203  * over current.
3204  * In the case of over temperature Marlin can decrease the drive until the
3205  * error condition clears.
3206  * Other detected conditions can be used to stop the current print.
3207  * Relevant G-codes:
3208  * M906 - I1/2/3/4/5 Set or get motor drive level using axis codes X, Y,
3209  * Z, E. Report values if no axis codes given.
3210  *           I not present or I0 or I1 - X, Y, Z or E0
3211  *           I2 - X2, Y2, Z2 or E1
3212  *           I3 - Z3 or E3
3213  *           I4 - Z4 or E4
3214  *           I5 - E5
3215  * M916 - Increase drive level until get thermal warning
3216  * M917 - Find minimum current thresholds
3217  * M918 - Increase speed until max or error
3218  * M122 S0/1 - Report driver parameters

```

```

3216 */
3217 // #define MONITOR_L6470_DRIVER_STATUS
3218
3219 #if ENABLED(MONITOR_L6470_DRIVER_STATUS)
3220     #define KVAL_HOLD_STEP_DOWN    1
3221     // #define L6470_STOP_ON_ERROR
3222 #endif
3223
3224#endif // HAS_L64XX
3225
3226// @section i2cbus
3227
3228//
3229// I2C Master ID for LPC176x LCD and Digital Current control
3230// Does not apply to other peripherals based on the Wire library.
3231//
3232// #define I2C_MASTER_ID 1 // Set a value from 0 to 2
3233
3234/**
3235 * TWI/I2C BUS
3236 *
3237 * This feature is an EXPERIMENTAL feature so it shall not be used on
3238 * production
3239 * machines. Enabling this will allow you to send and receive I2C data from
3240 * slave
3241 * devices on the bus.
3242 *
3243 * ; Example #1
3244 * ; This macro send the string "Marlin" to the slave device with address
3245 * 0x63 (99)
3246 * ; It uses multiple M260 commands with one B<base 10> arg
3247 * M260 A99 ; Target slave address
3248 * M260 B77 ; M
3249 * M260 B97 ; a
3250 * M260 B114 ; r
3251 * M260 B108 ; l
3252 * M260 B105 ; i
3253 * M260 B110 ; n
3254 * M260 S1 ; Send the current buffer
3255 *
3256 * ; Example #2
3257 * ; Request 6 bytes from slave device with address 0x63 (99)
3258 * M261 A99 B5
3259 *
3260 * ; Example #3
3261 * ; Example serial output of a M261 request
3262 * echo:i2c-reply: from:99 bytes:5 data:hello
3263 */
3264
3265 // #define EXPERIMENTAL_I2CBUS
3266 #if ENABLED(EXPERIMENTAL_I2CBUS)
3267     #define I2C_SLAVE_ADDRESS 0 // Set a value from 8 to 127 to act as a
3268     slave
3269 #endif
3270
3271// @section extras
3272
3273 /**
3274 * Dphoto C-code

```

```

3271 * PHOTO_GCODE
3272 * Add the M240 G-code to take a photo.
3273 * The photo can be triggered by a digital pin or a physical movement.
3274 */
3275 //#define PHOTO_GCODE
3276 #if ENABLED(PHOTO_GCODE)
3277     // A position to move to (and raise Z) before taking the photo
3278     //#define PHOTO_POSITION { X_MAX_POS - 5, Y_MAX_POS, 0 } // { xpos, ypos,
3279     zraise } (M240 X Y Z)
3280     //#define PHOTO_DELAY_MS    100                                // (ms) Duration
3281     to pause before moving back (M240 P)
3282     //#define PHOTO_RETRACT_MM   6.5                               // (mm) E
3283     retract/recover for the photo move (M240 R S)
3284
3285     // Canon RC-1 or homebrew digital camera trigger
3286     // Data from: https://www.doc-diy.net/photo/rc-1\_hacked/
3287     //#define PHOTOGRAPH_PIN 23
3288
3289     // Canon Hack Development Kit
3290     // https://captain-slow.dk/2014/03/09/3d-printing-timelapses/
3291     //#define CHDK_PIN        4
3292
3293     // Optional second move with delay to trigger the camera shutter
3294     //#define PHOTO_SWITCH_POSITION { X_MAX_POS, Y_MAX_POS } // { xpos, ypos
3295 } (M240 I J)
3296
3297     // Duration to hold the switch or keep CHDK_PIN high
3298     //#define PHOTO_SWITCH_MS   50 // (ms) (M240 D)
3299
3300 /**
3301 * PHOTO_PULSES_US may need adjustment depending on board and camera
3302 model.
3303 * Pin must be running at 48.4kHz.
3304 * Be sure to use a PHOTOGRAPH_PIN which can rise and fall quick enough.
3305 * (e.g., MKS SBase temp sensor pin was too slow, so used P1.23 on J8.)
3306 *
3307 * Example pulse data for Nikon: https://bit.ly/2FKD0Aq
3308 * IR Wiring: https://git.io/JvJf7
3309 */
3310 //#define PHOTO_PULSES_US { 2000, 27850, 400, 1580, 400, 3580, 400 } // // (μs) Durations for each 48.4kHz oscillation
3311 #ifdef PHOTO_PULSES_US
3312     #define PHOTO_PULSE_DELAY_US 13 // (μs) Approximate duration of each
3313 HIGH and LOW pulse in the oscillation
3314 #endif
3315 #endif
3316
3317 /**
3318 * Spindle & Laser control
3319 *
3320 * Add the M3, M4, and M5 commands to turn the spindle/laser on and off, and
3321 * to set spindle speed, spindle direction, and laser power.
3322 *
3323 * SuperPid is a router/spindle speed controller used in the CNC milling
3324 community.
3325 * Marlin can be used to turn the spindle on and off. It can also be used to
3326 set
3327 * the spindle speed from 5,000 to 30,000 RPM.
3328 *
3329

```

```

3320 * You'll need to select a pin for the ON/OFF function and optionally choose
3321 a 0-5V
3322 * hardware PWM pin for the speed control and a pin for the rotation
3323 direction.
3324 */
3325 //#define SPINDLE_FEATURE
3326 //#define LASER_FEATURE
3327 #if EITHER(SPINDLE_FEATURE, LASER_FEATURE)
3328   #define SPINDLE_LASER_ACTIVE_STATE    LOW      // Set to "HIGH" if
3329   SPINDLE_LASER_ENA_PIN is active HIGH
3330
3331   #define SPINDLE_LASER_USE_PWM          // Enable if your controller
3332 supports setting the speed/power
3333   #if ENABLED(SPINDLE_LASER_USE_PWM)
3334     #define SPINDLE_LASER_PWM_INVERT    false    // Set to "true" if the
3335 speed/power goes up when you want it to go slower
3336     #define SPINDLE_LASER_FREQUENCY    2500    // (Hz) Spindle/laser
3337 frequency (only on supported HALs: AVR and LPC)
3338   #endif
3339
3340   ///#define AIR_EVACUATION           // Cutter Vacuum / Laser
3341 Blower motor control with G-codes M10-M11
3342   #if ENABLED(AIR_EVACUATION)
3343     #define AIR_EVACUATION_ACTIVE      LOW      // Set to "HIGH" if the
3344 on/off function is active HIGH
3345     ///#define AIR_EVACUATION_PIN      42       // Override the default
3346 Cutter Vacuum or Laser Blower pin
3347   #endif
3348
3349   ///#define AIR_ASSIST              // Air Assist control with G-
3350 codes M8-M9
3351   #if ENABLED(AIR_ASSIST)
3352     #define AIR_ASSIST_ACTIVE        LOW      // Active state on air assist
3353     pin
3354     ///#define AIR_ASSIST_PIN        44       // Override the default Air
3355 Assist pin
3356   #endif
3357
3358   ///#define SPINDLE_SERVO           // A servo converting an
3359 angle to spindle power
3360   #ifdef SPINDLE_SERVO
3361     #define SPINDLE_SERVO_NR        0       // Index of servo used for
3362 spindle control
3363     #define SPINDLE_SERVO_MIN      10      // Minimum angle for servo
3364 spindle
3365   #endif
3366
3367 /**
3368  * Speed / Power can be set ('M3 S') and displayed in terms of:
3369  * - PWM255 (S0 - S255)
3370  * - PERCENT (S0 - S100)
3371  * - RPM      (S0 - S50000) Best for use with a spindle
3372  * - SERVO    (S0 - S180)
3373  */
3374 #define CUTTER_POWER_UNIT PWM255

```

```

3362
3363 /**
3364 * Relative Cutter Power
3365 * Normally, 'M3 0<power>' sets
3366 * OCR power is relative to the range SPEED_POWER_MIN...SPEED_POWER_MAX.
3367 * so input powers of 0...255 correspond to
3368 SPEED_POWER_MIN...SPEED_POWER_MAX
3369 * instead of normal range (0 to SPEED_POWER_MAX).
3370 * Best used with (e.g.) SuperPID router controller: S0 = 5,000 RPM and
3371 S255 = 30,000 RPM
3372 */
3373 // #define CUTTER_POWER_RELATIVE // Set speed proportional to
3374 [SPEED_POWER_MIN...SPEED_POWER_MAX]
3375
3376 #if ENABLED(SPINDLE_FEATURE)
3377 // #define SPINDLE_CHANGE_DIR // Enable if your spindle
3378 controller can change spindle direction
3379 #define SPINDLE_CHANGE_DIR_STOP // Enable if the spindle
3380 should stop before changing spin direction
3381 #define SPINDLE_INVERT_DIR false // Set to "true" if the spin
3382 direction is reversed
3383
3384 #define SPINDLE_LASER_POWERUP_DELAY 5000 // (ms) Delay to allow the
3385 spindle/laser to come up to speed/power
3386 #define SPINDLE_LASER_POWERDOWN_DELAY 5000 // (ms) Delay to allow the
3387 spindle to stop
3388
3389 /**
3390 * M3/M4 Power Equation
3391 *
3392 * Each tool uses different value ranges for speed / power control.
3393 * These parameters are used to convert between tool power units and
3394 PWM.
3395 *
3396 * Speed/Power = (PWMDC / 255 * 100 - SPEED_POWER_INTERCEPT) /
3397 SPEED_POWER_SLOPE
3398 * PWMDC = (spdpwr - SPEED_POWER_MIN) / (SPEED_POWER_MAX -
3399 SPEED_POWER_MIN) / SPEED_POWER_SLOPE
3400 */
3401 #if ENABLED(SPINDLE_LASER_USE_PWM)
3402 #define SPEED_POWER_INTERCEPT 0 // (%) 0-100 i.e., Minimum
3403 power percentage
3404 #define SPEED_POWER_MIN 5000 // (RPM)
3405 #define SPEED_POWER_MAX 30000 // (RPM) SuperPID router
3406 controller 0 - 30,000 RPM
3407 #define SPEED_POWER_STARTUP 25000 // (RPM) M3/M4 speed/power
3408 default (with no arguments)
3409 #endif
3410
3411 #else
3412
3413 #if ENABLED(SPINDLE_LASER_USE_PWM)
3414 #define SPEED_POWER_INTERCEPT 0 // (%) 0-100 i.e., Minimum
3415 power percentage
3416 #define SPEED_POWER_MIN 0 // (%) 0-100
3417 #define SPEED_POWER_MAX 100 // (%) 0-100
3418 #define SPEED_POWER_STARTUP 80 // (%) M3/M4 speed/power
3419 default (with no arguments)
3420 #endif

```

```

3405
3406 // Define the minimum and maximum test pulse time values for a laser
3407 test fire function
3408 #define LASER_TEST_PULSE_MIN           1 // Used with Laser Control
3409 Menu
3410 #define LASER_TEST_PULSE_MAX         999 // Caution: Menu may not show
3411 more than 3 characters
3412
3413 /**
3414  * Enable inline laser power to be handled in the planner / stepper
3415 routines.
3416  * Inline power is specified by the I (inline) flag in an M3 command
3417 (e.g., M3 S20 I)
3418  * or by the 'S' parameter in G0/G1/G2/G3 moves (see LASER_MOVE_POWER).
3419  *
3420  * This allows the laser to keep in perfect sync with the planner and
3421 removes
3422  * the powerup/down delay since lasers require negligible time.
3423  */
3424 //#define LASER_POWER_INLINE
3425
3426 #if ENABLED(LASER_POWER_INLINE)
3427 /**
3428  * Scale the laser's power in proportion to the movement rate.
3429  *
3430  * - Sets the entry power proportional to the entry speed over the
3431 nominal speed.
3432  * - Ramps the power up every N steps to approximate the speed
3433 trapezoid.
3434  * - Due to the limited power resolution this is only approximate.
3435  */
3436 #define LASER_POWER_INLINE_TRAPEZOID
3437
3438 /**
3439  * Continuously calculate the current power (nominal_power *
3440 current_rate / nominal_rate).
3441  * Required for accurate power with non-trapezoidal acceleration
3442 (e.g., S_CURVE_ACCELERATION).
3443  * This is a costly calculation so this option is discouraged on 8-bit
3444 AVR boards.
3445  *
3446  * LASER_POWER_INLINE_TRAPEZOID_CONT_PER defines how many step cycles
3447 there are between power updates. If your
3448  * board isn't able to generate steps fast enough (and you are using
3449 LASER_POWER_INLINE_TRAPEZOID_CONT), increase this.
3450  * Note that when this is zero it means it occurs every cycle; 1 means
3451 a delay wait one cycle then run, etc.
3452  */
3453 //#define LASER_POWER_INLINE_TRAPEZOID_CONT
3454
3455 /**
3456  * Stepper iterations between power updates. Increase this value if
3457 the board
3458  * can't keep up with the processing demands of
3459 LASER_POWER_INLINE_TRAPEZOID_CONT.
3460  * Disable (or set to 0) to recalculate power on every stepper
3461 iteration.
3462  */
3463 //#define LASER_POWER_INLINE_TRAPEZOID_CONT_DEF 10

```

```

3440 // #define LASER_POWER_INLINE_TRAPEZOID_CONT_PEN to
3441
3442 /**
3443  * Include laser power in G0/G1/G2/G3/G5 commands with the 'S'
3444 parameter
3445 */
3446 // #define LASER_MOVE_POWER
3447
3448 #if ENABLED(LASER_MOVE_POWER)
3449     // Turn off the laser on G0 moves with no power parameter.
3450     // If a power parameter is provided, use that instead.
3451     // #define LASER_MOVE_G0_OFF
3452
3453     // Turn off the laser on G28 homing.
3454     // #define LASER_MOVE_G28_OFF
3455 #endif
3456
3457 /**
3458  * Inline flag inverted
3459  *
3460  * WARNING: M5 will NOT turn off the laser unless another move
3461  *           is done (so G-code files must end with 'M5 I').
3462  */
3463 // #define LASER_POWER_INLINE_INVERT
3464
3465 /**
3466  * Continuously apply inline power. ('M3 S3' == 'G1 S3' == 'M3 S3 I')
3467  *
3468  * The laser might do some weird things, so only enable this
3469  * feature if you understand the implications.
3470  */
3471 // #define LASER_POWER_INLINE_CONTINUOUS
3472
3473 #else
3474
3475     #define SPINDLE_LASER_POWERUP_DELAY      50 // (ms) Delay to allow the
3476 spindle/laser to come up to speed/power
3477     #define SPINDLE_LASER_POWERDOWN_DELAY    50 // (ms) Delay to allow the
3478 spindle to stop
3479
3480 #endif
3481
3482 /**
3483  * Laser I2C Ammeter (High precision INA226 low/high side module)
3484  */
3485 // #define I2C_AMMETER
3486 #if ENABLED(I2C_AMMETER)
3487     #define I2C_AMMETER_IMAX              0.1    // (Amps) Calibration value
3488 for the expected current range
3489     #define I2C_AMMETER_SHUNT_RESISTOR   0.1    // (Ohms) Calibration shunt
3490 resistor value
3491 #endif
3492
3493 #endif
3494 #endif // SPINDLE_FEATURE || LASER_FEATURE
3495
3496 /**
3497  * Synchronous Laser Control with M106/M107
3498  *

```

```

3500 * Marlin normally applies M106/M107 fan speeds at a time "soon after"
3501 processing
3502 * a planner block. This is too inaccurate for a PWM/TTL laser attached to
3503 the fan
3504 * header (as with some add-on laser kits). Enable this option to set
3505 fan/laser
3506 * speeds with much more exact timing for improved print fidelity.
3507 */
3508 // #define LASER_SYNCHRONOUS_M106_M107
3509 /**
3510 * Coolant Control
3511 *
3512 * Add the M7, M8, and M9 commands to turn mist or flood coolant on and off.
3513 *
3514 * Note: COOLANT_MIST_PIN and/or COOLANT_FLOOD_PIN must also be defined.
3515 */
3516 // #define COOLANT_CONTROL
3517 #if ENABLED(COOLANT_CONTROL)
3518 #define COOLANT_MIST           // Enable if mist coolant is present
3519 #define COOLANT_FLOOD          // Enable if flood coolant is present
3520 #define COOLANT_MIST_INVERT    false // Set "true" if the on/off function
3521 is reversed
3522 #define COOLANT_FLOOD_INVERT  false // Set "true" if the on/off function
3523 is reversed
3524 #endif
3525 /**
3526 * Filament Width Sensor
3527 *
3528 * Measures the filament width in real-time and adjusts
3529 * flow rate to compensate for any irregularities.
3530 *
3531 * Also allows the measured filament diameter to set the
3532 * extrusion rate, so the slicer only has to specify the
3533 * volume.
3534 *
3535 * Only a single extruder is supported at this time.
3536 *
3537 * 34 RAMPS_14      : Analog input 5 on the AUX2 connector
3538 * 81 PRINTRBOARD  : Analog input 2 on the Exp1 connector (version B,C,D,E)
3539 * 301 RAMBO       : Analog input 3
3540 *
3541 * Note: May require analog pins to be defined for other boards.
3542 */
3543 // #define FILAMENT_WIDTH_SENSOR
3544 #if ENABLED(FILAMENT_WIDTH_SENSOR)
3545   #define FILAMENT_SENSOR_EXTRUDER_NUM 0 // Index of the extruder that
3546 has the filament sensor. : [0,1,2,3,4]
3547   #define MEASUREMENT_DELAY_CM        14 // (cm) The distance from the
3548 filament sensor to the melting chamber
3549   #define FILWIDTH_ERROR_MARGIN       1.0 // (mm) If a measurement differs
3550 too much from nominal width ignore it
3551   #define MAX_MEASUREMENT_DELAY     20 // (bytes) Buffer size for

```

```
3550 stored measurements (1 byte per cm). Must be larger than
3551 MEASUREMENT_DELAY_CM.

3552 #define DEFAULT_MEASURED_FILAMENT_DIA DEFAULT_NOMINAL_FILAMENT_DIA // Set
3553 measured to nominal initially

3554 // Display filament width on the LCD status line. Status messages will
3555 expire after 5 seconds.
3556 // #define FILAMENT_LCD_DISPLAY
3557 #endif

3558 /**
3559 * Power Monitor
3560 * Monitor voltage (V) and/or current (A), and –when possible– power (W)
3561 *
3562 * Read and configure with M430
3563 *
3564 * The current sensor feeds DC voltage (relative to the measured current) to
3565 * an analog pin
3566 * The voltage sensor feeds DC voltage (relative to the measured voltage) to
3567 * an analog pin
3568 */
3569 // #define POWER_MONITOR_CURRENT // Monitor the system current
3570 // #define POWER_MONITOR_VOLTAGE // Monitor the system voltage

3571 #if ENABLED(POWER_MONITOR_CURRENT)
3572     #define POWER_MONITOR_VOLTS_PER_AMP    0.05000 // Input voltage to the
3573     MCU analog pin per amp – DO NOT apply more than ADC_VREF!
3574     #define POWER_MONITOR_CURRENT_OFFSET   0          // Offset (in amps)
3575     applied to the calculated current
3576     #define POWER_MONITOR_FIXED_VOLTAGE  13.6        // Voltage for a current
3577     sensor with no voltage sensor (for power display)
3578 #endif

3579 #if ENABLED(POWER_MONITOR_VOLTAGE)
3580     #define POWER_MONITOR_VOLTS_PER_VOLT  0.077933 // Input voltage to the
3581     MCU analog pin per volt – DO NOT apply more than ADC_VREF!
3582     #define POWER_MONITOR_VOLTAGE_OFFSET  0          // Offset (in volts)
3583     applied to the calculated voltage
3584 #endif

3585 /**
3586 * Stepper Driver Anti-SNAFU Protection
3587 *
3588 * If the SAFE_POWER_PIN is defined for your board, Marlin will check
3589 * that stepper drivers are properly plugged in before applying power.
3590 * Disable protection if your stepper drivers don't support the feature.
3591 */
3592 // #define DISABLE_DRIVER_SAFE_POWER_PROTECT

3593 /**
3594 * CNC Coordinate Systems
3595 *
3596 * Enables G53 and G54–G59.3 commands to select coordinate systems
3597 * and G92.1 to reset the workspace to native machine space.
3598 */
3599 // #define CNC_COORDINATE_SYSTEMS

3600 /**

```

```
3598 * Auto-report temperatures with M155 S<seconds>
3599 */
3600 #define AUTO_REPORT_TEMPERATURES
3601
3602 /**
3603 * Auto-report position with M154 S<seconds>
3604 */
3605 // #define AUTO_REPORT_POSITION
3606
3607 /**
3608 * Include capabilities in M115 output
3609 */
3610 #define EXTENDED_CAPABILITIES_REPORT
3611 #if ENABLED(EXTENDED_CAPABILITIES_REPORT)
3612     // #define M115_GEOMETRY_REPORT
3613 #endif
3614
3615 /**
3616 * Expected Printer Check
3617 * Add the M16 G-code to compare a string to the MACHINE_NAME.
3618 * M16 with a non-matching string causes the printer to halt.
3619 */
3620 // #define EXPECTED_PRINTER_CHECK
3621
3622 /**
3623 * Disable all Volumetric extrusion options
3624 */
3625 // #define NO_VOLUMETRICS
3626
3627 #if DISABLED(NO_VOLUMETRICS)
3628 /**
3629 * Volumetric extrusion default state
3630 * Activate to make volumetric extrusion the default method,
3631 * with DEFAULT_NOMINAL_FILAMENT_DIA as the default diameter.
3632 *
3633 * M200 D0 to disable, M200 Dn to set a new diameter (and enable
3634 volumetric).
3635 * M200 S0/S1 to disable/enable volumetric extrusion.
3636 */
3637 // #define VOLUMETRIC_DEFAULT_ON
3638
3639 // #define VOLUMETRIC_EXTRUDER_LIMIT
3640 #if ENABLED(VOLUMETRIC_EXTRUDER_LIMIT)
3641 /**
3642 * Default volumetric extrusion limit in cubic mm per second (mm^3/sec).
3643 * This factory setting applies to all extruders.
3644 * Use 'M200 [T<extruder>] L<limit>' to override and 'M502' to reset.
3645 * A non-zero value activates Volume-based Extrusion Limiting.
3646 */
3647 #define DEFAULT_VOLUMETRIC_EXTRUDER_LIMIT 0.00      // (mm^3/sec)
3648#endif
3649
3650 /**
3651 * Enable this option for a leaner build of Marlin that removes all
3652 * workspace offsets, simplifying coordinate transformations, leveling, etc.
3653 *
3654 * - M206 and M428 are disabled.
3655 */
```

```
3655 * - G92 will revert to its behavior from Marlin 1.0.
3656 */
3657 //#define NO_WORKSPACE_OFFSETS
3658
3659 // Extra options for the M114 "Current Position" report
3660 //#define M114_DETAIL           // Use 'M114` for details to check planner
3661 calculations
3662 //#define M114_REALTIME        // Real current position based on forward
3663 kinematics
3664 //#define M114_LEGACY          // M114 used to synchronize on every call.
3665 Enable if needed.
3666
3667 /**
3668 * Set the number of proportional font spaces required to fill up a typical
3669 character space.
3670 * This can help to better align the output of commands like `G29 0` Mesh
3671 Output.
3672 *
3673 * For clients that use a fixed-width font (like OctoPrint), leave this set
3674 to 1.0.
3675 * Otherwise, adjust according to your client and font.
3676 */
3677 #define PROPORTIONAL_FONT_RATIO 1.0
3678
3679 /**
3680 * Spend 28 bytes of SRAM to optimize the G-code parser
3681 */
3682 #define FASTER_GCODE_PARSER
3683
3684 #if ENABLED(FASTER_GCODE_PARSER)
3685     //#define GCODE_QUOTED_STRINGS // Support for quoted string parameters
3686 #endif
3687
3688 // Support for MeatPack G-code compression
3689 // (https://github.com/scottmudge/OctoPrint-MeatPack)
3690 //#define MEATPACK_ON_SERIAL_PORT_1
3691 //#define MEATPACK_ON_SERIAL_PORT_2
3692
3693 /**
3694 * CNC G-code options
3695 * Support CNC-style G-code dialects used by laser cutters, drawing machine
3696 cams, etc.
3697 * Note that G0 feedrates should be used with care for 3D printing (if used
3698 at all).
3699 * High feedrates may cause ringing and harm print quality.
3700 */
3701
3702 /**
3703 * Support for parentheses-delimited comments
3704 */
3705
```

```
3701 // Enable and set a (default) feedrate for all G0 moves
3702 //#define G0_FEEDRATE 3000 // (mm/min)
3703 #ifdef G0_FEEDRATE
3704     //##define VARIABLE_G0_FEEDRATE // The G0 feedrate is set by F in G0 motion
3705     mode
3706 #endif
3707 /**
3708 * Startup commands
3709 *
3710 * Execute certain G-code commands immediately after power-on.
3711 */
3712 //#define STARTUP_COMMANDS "M17 Z"
3713
3714 /**
3715 * G-code Macros
3716 *
3717 * Add G-codes M810-M819 to define and run G-code macros.
3718 * Macros are not saved to EEPROM.
3719 */
3720 //#define GCODE_MACROS
3721 #if ENABLED(GCODE_MACROS)
3722     #define GCODE_MACROS_SLOTS      5 // Up to 10 may be used
3723     #define GCODE_MACROS_SLOT_SIZE 50 // Maximum length of a single macro
3724 #endif
3725
3726 /**
3727 * User-defined menu items to run custom G-code.
3728 * Up to 25 may be defined, but the actual number is LCD-dependent.
3729 */
3730
3731 // Custom Menu: Main Menu
3732 //#define CUSTOM_MENU_MAIN
3733 #if ENABLED(CUSTOM_MENU_MAIN)
3734     //##define CUSTOM_MENU_MAIN_TITLE "Custom Commands"
3735     #define CUSTOM_MENU_MAIN_SCRIPT_DONE "M117 User Script Done"
3736     #define CUSTOM_MENU_MAIN_SCRIPT_AUDIBLE_FEEDBACK
3737     //##define CUSTOM_MENU_MAIN_SCRIPT_RETURN // Return to status screen
3738     after a script
3739     #define CUSTOM_MENU_MAIN_ONLY_IDLE // Only show custom menu when
3740     the machine is idle
3741
3742     #define MAIN_MENU_ITEM_1_DESC "Home & UBL Info"
3743     #define MAIN_MENU_ITEM_1_GCODE "G28\\nG29 W"
3744     //##define MAIN_MENU_ITEM_1_CONFIRM // Show a confirmation dialog
3745     before this action
3746
3747     #define MAIN_MENU_ITEM_2_DESC "Preheat for " PREHEAT_1_LABEL
3748     #define MAIN_MENU_ITEM_2_GCODE "M140 S" STRINGIFY(PREHEAT_1_TEMP_BED)
3749     "\\nM104 S" STRINGIFY(PREHEAT_1_TEMP_HOTEND)
3750     //##define MAIN_MENU_ITEM_2_CONFIRM
3751
3752     //##define MAIN_MENU_ITEM_3_DESC "Preheat for " PREHEAT_2_LABEL
3753     //##define MAIN_MENU_ITEM_3_GCODE "M140 S" STRINGIFY(PREHEAT_2_TEMP_BED)
3754     "\\nM104 S" STRINGIFY(PREHEAT_2_TEMP_HOTEND)
3755     //##define MAIN_MENU_ITEM_3_CONFIRM
3756
3757     //##define MAIN_MENU_ITEM_4_DESC "Heat Bed/Home/Level"
3758     //...  
.....
```

```

3753 // #define MAIN_MENU_ITEM_4_GCODE "M140 S" STRINGIFY(PREHEAT_Z_TEMP_BED)
3754 " \nG28\nG29"
3755 // #define MAIN_MENU_ITEM_4_CONFIRM
3756 // #define MAIN_MENU_ITEM_5_DESC "Home & Info"
3757 // #define MAIN_MENU_ITEM_5_GCODE "G28\nM503"
3758 // #define MAIN_MENU_ITEM_5_CONFIRM
3759 #endif
3760
3761 // Custom Menu: Configuration Menu
3762 // #define CUSTOM_MENU_CONFIG
3763 #if ENABLED(CUSTOM_MENU_CONFIG)
3764 // #define CUSTOM_MENU_CONFIG_TITLE "Custom Commands"
3765 #define CUSTOM_MENU_CONFIG_SCRIPT_DONE "M117 Wireless Script Done"
3766 #define CUSTOM_MENU_CONFIG_SCRIPT_AUDIBLE_FEEDBACK
3767 // #define CUSTOM_MENU_CONFIG_SCRIPT_RETURN // Return to status screen
3768 after a script
3769 #define CUSTOM_MENU_CONFIG_ONLY_IDLE // Only show custom menu when
3770 the machine is idle
3771
3772 #define CONFIG_MENU_ITEM_1_DESC "Wifi ON"
3773 #define CONFIG_MENU_ITEM_1_GCODE "M118 [ESP110] WIFI-STA pwd=12345678"
3774 // #define CONFIG_MENU_ITEM_1_CONFIRM // Show a confirmation dialog
3775 before this action
3776
3777 #define CONFIG_MENU_ITEM_2_DESC "Bluetooth ON"
3778 #define CONFIG_MENU_ITEM_2_GCODE "M118 [ESP110] BT pwd=12345678"
3779 // #define CONFIG_MENU_ITEM_2_CONFIRM
3780
3781 // #define CONFIG_MENU_ITEM_3_DESC "Radio OFF"
3782 // #define CONFIG_MENU_ITEM_3_GCODE "M118 [ESP110] OFF pwd=12345678"
3783 // #define CONFIG_MENU_ITEM_3_CONFIRM
3784
3785 // #define CONFIG_MENU_ITEM_4_DESC "Wifi ????"
3786 // #define CONFIG_MENU_ITEM_4_GCODE "M118 ????"
3787 // #define CONFIG_MENU_ITEM_4_CONFIRM
3788
3789 #endif
3790
3791 /**
3792 * User-defined buttons to run custom G-code.
3793 * Up to 25 may be defined.
3794 */
3795 // #define CUSTOM_USER_BUTTONS
3796 #if ENABLED(CUSTOM_USER_BUTTONS)
3797 // #define BUTTON1_PIN -1
3798 #if PIN_EXISTS(BUTTON1)
3799     #define BUTTON1_HIT_STATE LOW // State of the triggered
3800 button. NC=LOW, NO=HIGH.
3801     #define BUTTON1_WHEN_PRINTING false // Button allowed to trigger
3802 during printing?
3803     #define BUTTON1_GCODE "G28"
3804     #define BUTTON1_DESC "Homing" // Optional string to set the
LCD status
3805 #endif

```

```

3805 //##define BUTTON2_PIN -1
3806 #if PIN_EXISTS(BUTTON2)
3807     #define BUTTON2_HIT_STATE      LOW
3808     #define BUTTON2_WHEN_PRINTING false
3809     #define BUTTON2_GCODE          "M140 S" STRINGIFY(PREHEAT_1_TEMP_BED)
3810     "\nM104 S" STRINGIFY(PREHEAT_1_TEMP_HOTEND)
3811     #define BUTTON2_DESC           "Preheat for " PREHEAT_1_LABEL
3812 #endif
3813
3814 //##define BUTTON3_PIN -1
3815 #if PIN_EXISTS(BUTTON3)
3816     #define BUTTON3_HIT_STATE      LOW
3817     #define BUTTON3_WHEN_PRINTING false
3818     #define BUTTON3_GCODE          "M140 S" STRINGIFY(PREHEAT_2_TEMP_BED)
3819     "\nM104 S" STRINGIFY(PREHEAT_2_TEMP_HOTEND)
3820     #define BUTTON3_DESC           "Preheat for " PREHEAT_2_LABEL
3821 #endif
3822 #endif
3823
3824 /**
3825 * Host Action Commands
3826 *
3827 * Define host streamer action commands in compliance with the standard.
3828 *
3829 * See https://reprap.org/wiki/G-code#Action\_commands
3830 * Common commands ..... poweroff, pause, paused, resume, resumed, cancel
3831 * G29_RETRY_AND_RECOVER .. probe_rewipe, probe_failed
3832 *
3833 * Some features add reason codes to extend these commands.
3834 *
3835 * Host Prompt Support enables Marlin to use the host for user prompts so
3836 * filament runout and other processes can be managed from the host side.
3837 */
3838 //##define HOST_ACTION_COMMANDS
3839 #if ENABLED(HOST_ACTION_COMMANDS)
3840     //##define HOST_PAUSE_M76
3841     //##define HOST_PROMPT_SUPPORT
3842     //##define HOST_START_MENU_ITEM // Add a menu item that tells the host to
3843     start
3844 #endif
3845
3846 /**
3847 * Cancel Objects
3848 *
3849 * Implement M486 to allow Marlin to skip objects
3850 */
3851 //##define CANCEL_OBJECTS
3852 #if ENABLED(CANCEL_OBJECTS)
3853     #define CANCEL_OBJECTS_REPORTING // Emit the current object as a status
3854     message
3855 #endif
3856
3857 /**
3858 * I2C position encoders for closed loop control.
3859 * Developed by Chris Barr at Aus3D.
3860 *
3861 * Wiki: https://wiki.aus3d.com.au/Magnetic\_Encoder
3862 * Github: https://github.com/Aus3D/MagneticEncoder

```

```

3859 *
3860 * Supplier: https://aus3d.com.au/magnetic-encoder-module
3861 * Alternative Supplier: https://reliabuild3d.com/
3862 *
3863 * Reliabuild encoders have been modified to improve reliability.
3864 */
3865
3866 // #define I2C_POSITION_ENCODERS
3867 #if ENABLED(I2C_POSITION_ENCODERS)
3868
3869     #define I2CPE_ENCODER_CNT           1                                // The number of
3870     encoders installed; max of 5                                     // encoders
3871
3872     supported currently.
3873
3874     #define I2CPE_ENC_1_ADDR          I2CPE_PRESET_ADDR_X      // I2C address
3875     of the encoder. 30-200.
3876     #define I2CPE_ENC_1_AXIS          X_AXIS                         // Axis the
3877     encoder module is installed on. <X|Y|Z|E>_AXIS.
3878     #define I2CPE_ENC_1_TYPE          I2CPE_ENC_TYPE_LINEAR    // Type of
3879     encoder: I2CPE_ENC_TYPE_LINEAR -or-
3880
3881     // I2CPE_ENC_TYPE_ROTARY.
3882     #define I2CPE_ENC_1_TICKS_UNIT   2048                            // 1024 for
3883     magnetic strips with 2mm poles; 2048 for
3884
3885     // For linear encoders this is ticks / mm,
3886
3887     // encoders this is ticks / revolution.
3888     // #define I2CPE_ENC_1_TICKS_REV (16 * 200)                      // Only needed
3889     // for rotary encoders; number of stepper
3890
3891     // full revolution (motor steps/rev * microstepping)
3892     // #define I2CPE_ENC_1_INVERT
3893     // direction of axis travel.
3894     #define I2CPE_ENC_1_EC_METHOD    I2CPE_ECM_MICROSTEP      // Type of error
3895     error correction.
3896     #define I2CPE_ENC_1_EC_THRESH   0.10                            // Threshold
3897
3898     size for error (in mm) above which the
3899
3900     attempt to correct the error; errors
3901
3902     this are ignored to minimize effects of
3903
3904     noise / latency (filter).
3905
3906
3907     #define I2CPE_ENC_2_ADDR          I2CPE_PRESET_ADDR_Y      // Same as
3908     above, but for encoder 2.
3909     #define I2CPE_ENC_2_AXIS          Y_AXIS
3910     #define I2CPE_ENC_2_TYPE          I2CPE_ENC_TYPE_LINEAR
3911     #define I2CPE_ENC_2_TICKS_UNIT   2048
3912     // #define I2CPE_ENC_2_TICKS_REV (16 * 200)
3913     // #define I2CPE_ENC_2_INVERT
3914     #define I2CPE_ENC_2_EC_METHOD    I2CPE_ECM_MICROSTEP
3915     #define I2CPE_ENC_2_EC_THRESH   0.10
3916
3917
3918     #define I2CPE_ENC_3_ADDR          I2CPE_PRESET_ADDR_Z      // Encoder 3.
3919     Add additional configuration options
3920     #define T2CPF FNC 3 AXTS
3921
3922     7 AXTS
3923
3924     // as above, or

```

```

-----  

use defaults below.  

3899  

3900 #define I2CPE_ENC_4_ADDR           I2CPE_PRESET_ADDR_E      // Encoder 4.  

3901 #define I2CPE_ENC_4_AXIS          E_AXIS  

3902  

3903 #define I2CPE_ENC_5_ADDR           34                      // Encoder 5.  

3904 #define I2CPE_ENC_5_AXIS          E_AXIS  

3905  

3906 // Default settings for encoders which are enabled, but without settings  

3907 // configured above.  

3908 #define I2CPE_DEF_TYPE            I2CPE_ENC_TYPE_LINEAR  

3909 #define I2CPE_DEF_ENC_TICKS_UNIT  2048  

3910 #define I2CPE_DEF_TICKS_REV        (16 * 200)  

3911 #define I2CPE_DEF_EC_METHOD       I2CPE_ECM_NONE  

3912 #define I2CPE_DEF_EC_THRESH        0.1  

3913  

3914 //##define I2CPE_ERR_THRESH_ABORT 100.0                  // Threshold  

3915 // size for error (in mm) error on any given  

3916 // axis after  

3917 // which the printer will abort. Comment out to  

3918 // disable abort  

3919 behavior.  

3920  

3921 #define I2CPE_TIME_TRUSTED        10000                 // After an  

3922 encoder fault, there must be no further fault  

3923 // for this  

3924 amount of time (in ms) before the encoder  

3925 // is trusted  

3926 again.  

3927  

3928 /**  

3929  * Position is checked every time a new command is executed from the  

3930  buffer but during long moves,  

3931  * this setting determines the minimum update time between checks. A value  

3932  of 100 works well with  

3933  * error rolling average when attempting to correct only for skips and not  

3934  for vibration.  

3935  */  

3936 #define I2CPE_MIN_UPD_TIME_MS     4                     // (ms) Minimum  

3937 time between encoder checks.  

3938  

3939 // Use a rolling average to identify persistent errors that indicate  

3940 // skips, as opposed to vibration and noise.  

3941 #define I2CPE_ERR_ROLLING_AVERAGE  

3942  

3943 #endif // I2C_POSITION_ENCODERS  

3944  

3945 /**  

3946  * Analog Joystick(s)  

3947  */  

3948 //##define JOYSTICK  

3949 #if ENABLED(JOYSTICK)
3950  #define JOY_X_PIN    5 // RAMPS: Suggested pin A5 on AUX2
3951  #define JOY_Y_PIN    10 // RAMPS: Suggested pin A10 on AUX2
3952  #define JOY_Z_PIN    12 // RAMPS: Suggested pin A12 on AUX2
3953  #define JOY_EN_PIN   44 // RAMPS: Suggested pin D44 on AUX2
3954  

3955 //##define INVERT_JOY_X // Enable if X direction is reversed

```

```

3944 //##define INVERT_JOY_Y // Enable if Y direction is reversed
3945 //##define INVERT_JOY_Z // Enable if Z direction is reversed
3946
3947 // Use M119 with JOYSTICK_DEBUG to find reasonable values after
3948 // connecting:
3949 #define JOY_X_LIMITS { 5600, 8190-100, 8190+100, 10800 } // min, deadzone
3950 // start, deadzone end, max
3951 #define JOY_Y_LIMITS { 5600, 8250-100, 8250+100, 11000 }
3952 #define JOY_Z_LIMITS { 4800, 8080-100, 8080+100, 11550 }
3953 //##define JOYSTICK_DEBUG
3954 #endif
3955
3956 /**
3957 * Mechanical Gantry Calibration
3958 * Modern replacement for the Prusa TMC_Z_CALIBRATION.
3959 * Adds capability to work with any adjustable current drivers.
3960 * Implemented as G34 because M915 is deprecated.
3961 */
3962 //##define MECHANICAL_GANTRY_CALIBRATION
3963 #if ENABLED(MECHANICAL_GANTRY_CALIBRATION)
3964     #define GANTRY_CALIBRATION_CURRENT          600      // Default calibration
3965     current in ma
3966     #define GANTRY_CALIBRATION_EXTRA_HEIGHT      15       // Extra distance in
3967     mm past Z_###_POS to move
3968     #define GANTRY_CALIBRATION_FEEDRATE         500      // Feedrate for
3969     correction move
3970     //##define GANTRY_CALIBRATION_TO_MIN        // Enable to calibrate
3971     Z in the MIN direction
3972
3973     //##define GANTRY_CALIBRATION_SAFE_POSITION XY_CENTER // Safe position for
3974     nozzle
3975     //##define GANTRY_CALIBRATION_XY_PARK_FEEDRATE 3000 // XY Park Feedrate -
3976     MMM
3977     //##define GANTRY_CALIBRATION_COMMANDS_PRE    ""
3978     #define GANTRY_CALIBRATION_COMMANDS_POST   "G28"      // G28 highly
3979     recommended to ensure an accurate position
3980 #endif
3981
3982 /**
3983 * Instant freeze / unfreeze functionality
3984 * Specified pin has pullup and connecting to ground will instantly pause
3985 motion.
3986 * Potentially useful for emergency stop that allows being resumed.
3987 */
3988 //##define FREEZE_FEATURE
3989 #if ENABLED(FREEZE_FEATURE)
3990     //##define FREEZE_PIN 41 // Override the default (KILL) pin here
3991 #endif
3992
3993 /**
3994 * MAX7219 Debug Matrix
3995 *
3996 * Add support for a low-cost 8x8 LED Matrix based on the Max7219 chip as a
3997 realtime status display.
3998 * Requires 3 signal wires. Some useful debug options are included to
3999 demonstrate its usage.
4000 */
4001 //##define MAX7219_DEBUG
4002 #if FNARIFN(MAX7219_NFRIG)

```

```

3991 #define MAX7219_CLK_PIN    64
3992 #define MAX7219_DIN_PIN    57
3993 #define MAX7219_LOAD_PIN   44
3994
3995 //##define MAX7219_GCODE           // Add the M7219 G-code to control the
3996 LED matrix
3997 #define MAX7219_INIT_TEST     2 // Test pattern at startup: 0=none,
3998 1=sweep, 2=spiral
3999 #define MAX7219_NUMBER_UNITS 1 // Number of Max7219 units in chain.
4000 #define MAX7219_ROTATE       0 // Rotate the display clockwise (in
4001 multiples of +/- 90°)
4002                                     // connector at: right=0 bottom=-90
4003 top=90 left=180
4004 //##define MAX7219_REVERSE_ORDER // The individual LED matrix units may be
4005 in reversed order
4006 //##define MAX7219_SIDE_BY_SIDE // Big chip+matrix boards can be chained
4007 side-by-side
4008
4009 /**
4010 * Sample debug features
4011 * If you add more debug displays, be careful to avoid conflicts!
4012 */
4013 #define MAX7219_DEBUG_PRINTER_ALIVE // Blink corner LED of 8x8 matrix
4014 to show that the firmware is functioning
4015 #define MAX7219_DEBUG_PLANNER_HEAD 3 // Show the planner queue head
4016 position on this and the next LED matrix row
4017 #define MAX7219_DEBUG_PLANNER_TAIL 5 // Show the planner queue tail
4018 position on this and the next LED matrix row
4019
4020 #define MAX7219_DEBUG_PLANNER_QUEUE 0 // Show the current planner queue
4021 depth on this and the next LED matrix row
4022                                     // If you experience stuttering,
4023 reboots, etc. this option can reveal how
4024                                     // tweaks made to the configuration
4025 are affecting the printer in real-time.
4026#endif
4027
4028 /**
4029 * NanoDLP Sync support
4030 *
4031 * Support for Synchronized Z moves when used with NanoDLP. G0/G1 axis moves
4032 will
4033 * output a "Z_move_comp" string to enable synchronization with DLP
4034 projector exposure.
4035 * This feature allows you to use [[WaitForDoneMessage]] instead of M400
4036 commands.
4037 */
4038 //##define NANODLP_Z_SYNC
4039 #if ENABLED(NANODLP_Z_SYNC)
4040     //##define NANODLP_ALL_AXIS // Send a "Z_move_comp" report for any axis
4041 move (not just Z).
4042#endif
4043
4044 /**
4045 * Ethernet. Use M552 to enable and set the IP address.
4046 */
4047 #if HAS_ETHERNET
4048     #define MAC_ADDRESS { 0xDE, 0xAD, 0xBE, 0xEF, 0xF0, 0x0D } // A MAC

```

```

address unique to your network
4033 #endif
4034
4035 /**
4036 * WiFi Support (Espressif ESP32 WiFi)
4037 */
4038 // #define WIFI_SUPPORT           // Marlin embedded WiFi management
4039 // #define ESP3D_WIFI_SUPPORT    // ESP3D Library WiFi management
4040 (https://github.com/luc-github/ESP3DLib)
4041
4042 #if EITHER(WIFI_SUPPORT, ESP3D_WIFI_SUPPORT)
4043     // #define WEBSUPPORT          // Start a webserver (which may include
4044     // auto-discovery)
4045     // #define OTASUPPORT          // Support over-the-air firmware updates
4046     // #define WIFI_CUSTOM_COMMAND // Accept feature config commands (e.g.,
4047     // WiFi ESP3D) from the host
4048
4049 /**
4050 * To set a default WiFi SSID / Password, create a file called
4051 Configuration_Secure.h with
4052 * the following defines, customized for your network. This specific file
4053 is excluded via
4054 * .gitignore to prevent it from accidentally leaking to the public.
4055 *
4056 * #define WIFI_SSID "WiFi SSID"
4057 * #define WIFI_PWD  "WiFi Password"
4058 */
4059 // #include "Configuration_Secure.h" // External file with WiFi SSID /
4060 Password
4061 #endif
4062
4063 /**
4064 * Průša Multi-Material Unit (MMU)
4065 * Enable in Configuration.h
4066 *
4067 * These devices allow a single stepper driver on the board to drive
4068 * multi-material feeders with any number of stepper motors.
4069 */
4070 #if HAS_PRUSA_MMU1
4071 /**
4072 * This option only allows the multiplexer to switch on tool-change.
4073 * Additional options to configure custom E moves are pending.
4074 *
4075 * Override the default DIO selector pins here, if needed.
4076 * Some pins files may provide defaults for these pins.
4077 */
4078 // #define E_MUX0_PIN 40 // Always Required
4079 // #define E_MUX1_PIN 42 // Needed for 3 to 8 inputs
4080 // #define E_MUX2_PIN 44 // Needed for 5 to 8 inputs
4081 #elif HAS_PRUSA_MMU2
4082 // Serial port used for communication with MMU2.
4083 #define MMU2_SERIAL_PORT 2
4084
4085 // Use hardware reset for MMU if a pin is defined for it
4086 // #define MMU2_RST_PIN 23
4087
4088 // Enable if the MMU2 has 12V stepper motors (MMU2 Firmware 1.0.2 and up)
4089 // #define MMU2_MODE_12V
4090

```

```

4085 // G-code to execute when MMU2 F.I.N.D.A. probe detects filament runout
4086 #define MMU2_FILAMENT_RUNOUT_SCRIPT "M600"
4087
4088 // Add an LCD menu for MMU2
4089 // #define MMU2_MENUS
4090 #if EITHER(MMU2_MENUS, HAS_PRUSA_MMU2S)
4091     // Settings for filament load / unload from the LCD menu.
4092     // This is for Průša MK3-style extruders. Customize for your hardware.
4093     #define MMU2_FILAMENTCHANGE_EJECT_FEED 80.0
4094     #define MMU2_LOAD_TO_NOZZLE_SEQUENCE \
4095         { 7.2, 1145 }, \
4096         { 14.4, 871 }, \
4097         { 36.0, 1393 }, \
4098         { 14.4, 871 }, \
4099         { 50.0, 198 }
4100
4101     #define MMU2_RAMMING_SEQUENCE \
4102         { 1.0, 1000 }, \
4103         { 1.0, 1500 }, \
4104         { 2.0, 2000 }, \
4105         { 1.5, 3000 }, \
4106         { 2.5, 4000 }, \
4107         { -15.0, 5000 }, \
4108         { -14.0, 1200 }, \
4109         { -6.0, 600 }, \
4110         { 10.0, 700 }, \
4111         { -10.0, 400 }, \
4112         { -50.0, 2000 }
4113 #endif
4114
4115 /**
4116 * Using a sensor like the MMU2S
4117 * This mode requires a MK3S extruder with a sensor at the extruder idler,
4118 like the MMU2S.
4119 * See https://help.prusa3d.com/en/guide/3b-mk3s-mk2-5s-extruder-
4120 upgrade_41560, step 11
4121 */
4122 #if HAS_PRUSA_MMU2S
4123     #define MMU2_C0_RETRY      5           // Number of retries (total time =
4124                                         timeout*retries)
4125
4126     #define MMU2_CAN_LOAD_FEEDRATE 800      // (mm/min)
4127     #define MMU2_CAN_LOAD_SEQUENCE \
4128         { 0.1, MMU2_CAN_LOAD_FEEDRATE }, \
4129         { 60.0, MMU2_CAN_LOAD_FEEDRATE }, \
4130         { -52.0, MMU2_CAN_LOAD_FEEDRATE }
4131
4132     #define MMU2_CAN_LOAD_RETRACT    6.0    // (mm) Keep under the distance
4133 between Load Sequence values
4134     #define MMU2_CAN_LOAD_DEVIATION 0.8    // (mm) Acceptable deviation
4135
4136     #define MMU2_CAN_LOAD_INCREMENT 0.2    // (mm) To reuse within MMU2
4137 module
4138     #define MMU2_CAN_LOAD_INCREMENT_SEQUENCE \
4139         { -MMU2_CAN_LOAD_INCREMENT, MMU2_CAN_LOAD_FEEDRATE }
4140
4141 #else
4142
4143
4144
4145
4146
4147
4148
4149
4150
4151
4152
4153
4154
4155
4156
4157
4158
4159
4160
4161
4162
4163
4164
4165
4166
4167
4168
4169
4170
4171
4172
4173
4174
4175
4176
4177
4178
4179
4180
4181
4182
4183
4184
4185
4186
4187
4188
4189
4190
4191
4192
4193
4194
4195
4196
4197
4198
4199
4200
4201
4202
4203
4204
4205
4206
4207
4208
4209
4210
4211
4212
4213
4214
4215
4216
4217
4218
4219
4220
4221
4222
4223
4224
4225
4226
4227
4228
4229
4230
4231
4232
4233
4234
4235
4236
4237
4238
4239
4240
4241
4242
4243
4244
4245
4246
4247
4248
4249
4250
4251
4252
4253
4254
4255
4256
4257
4258
4259
4260
4261
4262
4263
4264
4265
4266
4267
4268
4269
4270
4271
4272
4273
4274
4275
4276
4277
4278
4279
4280
4281
4282
4283
4284
4285
4286
4287
4288
4289
4290
4291
4292
4293
4294
4295
4296
4297
4298
4299
4300
4301
4302
4303
4304
4305
4306
4307
4308
4309
4310
4311
4312
4313
4314
4315
4316
4317
4318
4319
4320
4321
4322
4323
4324
4325
4326
4327
4328
4329
4330
4331
4332
4333
4334
4335
4336
4337
4338
4339
4340
4341
4342
4343
4344
4345
4346
4347
4348
4349
4350
4351
4352
4353
4354
4355
4356
4357
4358
4359
4360
4361
4362
4363
4364
4365
4366
4367
4368
4369
4370
4371
4372
4373
4374
4375
4376
4377
4378
4379
4380
4381
4382
4383
4384
4385
4386
4387
4388
4389
4390
4391
4392
4393
4394
4395
4396
4397
4398
4399
4400
4401
4402
4403
4404
4405
4406
4407
4408
4409
4410
4411
4412
4413
4414
4415
4416
4417
4418
4419
4420
4421
4422
4423
4424
4425
4426
4427
4428
4429
4430
4431
4432
4433
4434
4435
4436
4437
4438
4439
4440
4441
4442
4443
4444
4445
4446
4447
4448
4449
4450
4451
4452
4453
4454
4455
4456
4457
4458
4459
4460
4461
4462
4463
4464
4465
4466
4467
4468
4469
4470
4471
4472
4473
4474
4475
4476
4477
4478
4479
4480
4481
4482
4483
4484
4485
4486
4487
4488
4489
4490
4491
4492
4493
4494
4495
4496
4497
4498
4499
4500
4501
4502
4503
4504
4505
4506
4507
4508
4509
4510
4511
4512
4513
4514
4515
4516
4517
4518
4519
4520
4521
4522
4523
4524
4525
4526
4527
4528
4529
4530
4531
4532
4533
4534
4535
4536
4537
4538
4539
4540
4541
4542
4543
4544
4545
4546
4547
4548
4549
4550
4551
4552
4553
4554
4555
4556
4557
4558
4559
4560
4561
4562
4563
4564
4565
4566
4567
4568
4569
4570
4571
4572
4573
4574
4575
4576
4577
4578
4579
4580
4581
4582
4583
4584
4585
4586
4587
4588
4589
4590
4591
4592
4593
4594
4595
4596
4597
4598
4599
4600
4601
4602
4603
4604
4605
4606
4607
4608
4609
4610
4611
4612
4613
4614
4615
4616
4617
4618
4619
4620
4621
4622
4623
4624
4625
4626
4627
4628
4629
4630
4631
4632
4633
4634
4635
4636
4637
4638
4639
4640
4641
4642
4643
4644
4645
4646
4647
4648
4649
4650
4651
4652
4653
4654
4655
4656
4657
4658
4659
4660
4661
4662
4663
4664
4665
4666
4667
4668
4669
4670
4671
4672
4673
4674
4675
4676
4677
4678
4679
4680
4681
4682
4683
4684
4685
4686
4687
4688
4689
4690
4691
4692
4693
4694
4695
4696
4697
4698
4699
4700
4701
4702
4703
4704
4705
4706
4707
4708
4709
4710
4711
4712
4713
4714
4715
4716
4717
4718
4719
4720
4721
4722
4723
4724
4725
4726
4727
4728
4729
4730
4731
4732
4733
4734
4735
4736
4737
4738
4739
4740
4741
4742
4743
4744
4745
4746
4747
4748
4749
4750
4751
4752
4753
4754
4755
4756
4757
4758
4759
4760
4761
4762
4763
4764
4765
4766
4767
4768
4769
4770
4771
4772
4773
4774
4775
4776
4777
4778
4779
4780
4781
4782
4783
4784
4785
4786
4787
4788
4789
4790
4791
4792
4793
4794
4795
4796
4797
4798
4799
4800
4801
4802
4803
4804
4805
4806
4807
4808
4809
4810
4811
4812
4813
4814
4815
4816
4817
4818
4819
4820
4821
4822
4823
4824
4825
4826
4827
4828
4829
4830
4831
4832
4833
4834
4835
4836
4837
4838
4839
4840
4841
4842
4843
4844
4845
4846
4847
4848
4849
4850
4851
4852
4853
4854
4855
4856
4857
4858
4859
4860
4861
4862
4863
4864
4865
4866
4867
4868
4869
4870
4871
4872
4873
4874
4875
4876
4877
4878
4879
4880
4881
4882
4883
4884
4885
4886
4887
4888
4889
4890
4891
4892
4893
4894
4895
4896
4897
4898
4899
4900
4901
4902
4903
4904
4905
4906
4907
4908
4909
4910
4911
4912
4913
4914
4915
4916
4917
4918
4919
4920
4921
4922
4923
4924
4925
4926
4927
4928
4929
4930
4931
4932
4933
4934
4935
4936
4937
4938
4939
4940
4941
4942
4943
4944
4945
4946
4947
4948
4949
4950
4951
4952
4953
4954
4955
4956
4957
4958
4959
4960
4961
4962
4963
4964
4965
4966
4967
4968
4969
4970
4971
4972
4973
4974
4975
4976
4977
4978
4979
4980
4981
4982
4983
4984
4985
4986
4987
4988
4989
4990
4991
4992
4993
4994
4995
4996
4997
4998
4999
5000
5001
5002
5003
5004
5005
5006
5007
5008
5009
5010
5011
5012
5013
5014
5015
5016
5017
5018
5019
5020
5021
5022
5023
5024
5025
5026
5027
5028
5029
5030
5031
5032
5033
5034
5035
5036
5037
5038
5039
5040
5041
5042
5043
5044
5045
5046
5047
5048
5049
5050
5051
5052
5053
5054
5055
5056
5057
5058
5059
5060
5061
5062
5063
5064
5065
5066
5067
5068
5069
5070
5071
5072
5073
5074
5075
5076
5077
5078
5079
5080
5081
5082
5083
5084
5085
5086
5087
5088
5089
5090
5091
5092
5093
5094
5095
5096
5097
5098
5099
5100
5101
5102
5103
5104
5105
5106
5107
5108
5109
5110
5111
5112
5113
5114
5115
5116
5117
5118
5119
5120
5121
5122
5123
5124
5125
5126
5127
5128
5129
5130
5131
5132
5133
5134
5135
5136
5137
5138
5139
5140
5141
5142
5143
5144
5145
5146
5147
5148
5149
5150
5151
5152
5153
5154
5155
5156
5157
5158
5159
5160
5161
5162
5163
5164
5165
5166
5167
5168
5169
5170
5171
5172
5173
5174
5175
5176
5177
5178
5179
5180
5181
5182
5183
5184
5185
5186
5187
5188
5189
5190
5191
5192
5193
5194
5195
5196
5197
5198
5199
5200
5201
5202
5203
5204
5205
5206
5207
5208
5209
5210
5211
5212
5213
5214
5215
5216
5217
5218
5219
5220
5221
5222
5223
5224
5225
5226
5227
5228
5229
5230
5231
5232
5233
5234
5235
5236
5237
5238
5239
5240
5241
5242
5243
5244
5245
5246
5247
5248
5249
5250
5251
5252
5253
5254
5255
5256
5257
5258
5259
5260
5261
5262
5263
5264
5265
5266
5267
5268
5269
5270
5271
5272
5273
5274
5275
5276
5277
5278
5279
5280
5281
5282
5283
5284
5285
5286
5287
5288
5289
5290
5291
5292
5293
5294
5295
5296
5297
5298
5299
5300
5301
5302
5303
5304
5305
5306
5307
5308
5309
5310
5311
5312
5313
5314
5315
5316
5317
5318
5319
5320
5321
5322
5323
5324
5325
5326
5327
5328
5329
5330
5331
5332
5333
5334
5335
5336
5337
5338
5339
5340
5341
5342
5343
5344
5345
5346
5347
5348
5349
5350
5351
5352
5353
5354
5355
5356
5357
5358
5359
5360
5361
5362
5363
5364
5365
5366
5367
5368
5369
5370
5371
5372
5373
5374
5375
5376
5377
5378
5379
5380
5381
5382
5383
5384
5385
5386
5387
5388
5389
5390
5391
5392
5393
5394
5395
5396
5397
5398
5399
5400
5401
5402
5403
5404
5405
5406
5407
5408
5409
5410
5411
5412
5413
5414
5415
5416
5417
5418
5419
5420
5421
5422
5423
5424
5425
5426
5427
5428
5429
5430
5431
5432
5433
5434
5435
5436
5437
5438
5439
5440
5441
5442
5443
5444
5445
5446
5447
5448
5449
5450
5451
5452
5453
5454
5455
5456
5457
5458
5459
5460
5461
5462
5463
5464
5465
5466
5467
5468
5469
5470
5471
5472
5473
5474
5475
5476
5477
5478
5479
5480
5481
5482
5483
5484
5485
5486
5487
5488
5489
5490
5491
5492
5493
5494
5495
5496
5497
5498
5499
5500
5501
5502
5503
5504
5505
5506
5507
5508
5509
5510
5511
5512
5513
5514
5515
5516
5517
5518
5519
5520
5521
5522
5523
5524
5525
5526
5527
5528
5529
5530
5531
5532
5533
5534
5535
5536
5537
5538
5539
5540
5541
5542
5543
5544
5545
5546
5547
5548
5549
5550
5551
5552
5553
5554
5555
5556
5557
5558
5559
5560
5561
5562
5563
5564
5565
5566
5567
5568
5569
5570
5571
5572
5573
5574
5575
5576
5577
5578
5579
5580
5581
5582
5583
5584
5585
5586
5587
5588
5589
5590
5591
5592
5593
5594
5595
5596
5597
5598
5599
5600
5601
5602
5603
5604
5605
5606
5607
5608
5609
5610
5611
5612
5613
5614
5615
5616
5617
5618
5619
5620
5621
5622
5623
5624
5625
5626
5627
5628
5629
5630
5631
5632
5633
5634
5635
5636
5637
5638
5639
5640
5641
5642
5643
5644
5645
5646
5647
5648
5649
5650
5651
5652
5653
5654
5655
5656
5657
5658
5659
5660
5661
5662
5663
5664
5665
5666
5667
5668
5669
5670
5671
5672
5673
5674
5675
5676
5677
5678
5679
5680
5681
5682
5683
5684
5685
5686
5687
5688
5689
5690
5691
5692
5693
5694
5695
5696
5697
5698
5699
5700
5701
5702
5703
5704
5705
5706
5707
5708
5709
5710
5711
5712
5713
5714
5715
5716
5717
5718
5719
5720
5721
5722
5723
5724
5725
5726
5727
5728
5729
5730
5731
5732
5733
5734
5735
5736
5737
5738
5739
5740
5741
5742
5743
5744
5745
5746
5747
5748
5749
5750
5751
5752
5753
5754
5755
5756
5757
5758
5759
5760
5761
5762
5763
5764
5765
5766
5767
5768
5769
5770
5771
5772
5773
5774
5775
5776
5777
5778
5779
5780
5781
5782
5783
5784
5785
5786
5787
5788
5789
5790
5791
5792
5793
5794
5795
5796
5797
5798
5799
5800
5801
5802
5803
5804
5805
5806
5807
5808
5809
5810
5811
5812
5813
5814
5815
5816
5817
5818
5819
5820
5821
5822
5823
5824
5825
5826
5827
5828
5829
5830
5831
5832
5833
5834
5835
5836
5837
5838
5839
5840
5841
5842
5843
5844
5845
5846
5847
5848
5849
5850
5851
5852
5853
5854
5855
5856
5857
5858
5859
5860
5861
5862
5863
5864
5865
5866
5867
5868
5869
5870
5871
5872
5873
5874
5875
5876
5877
5878
5879
5880
5881
5882
5883
5884
5885
5886
5887
5888
5889
5890
5891
5892
5893
5894
5895
5896
5897
5898
5899
5900
5901
5902
5903
5904
5905
5906
5907
5908
5909
5910
5911
5912
5913
5914
5915
5916
5917
5918
5919
5920
5921
5922
5923
5924
5925
5926
5927
5928
5929
5930
5931
5932
5933
5934
5935
5936
5937
5938
5939
5940
5941
5942
5943
5944
5945
5946
5947
5948
5949
5950
5951
5952
5953
5954
5955
5956
5957
5958
5959
5960
5961
5962
5963
5964
5965
5966
5967
5968
5969
5970
5971
5972
5973
5974
5975
5976
5977
5978
5979
5980
5981
5982
5983
5984
5985
5986
5987
5988
5989
5990
5991
5992
5993
5994
5995
5996
5997
5998
5999
5999

```

```

4138 /**
4139 * MMU1 Extruder Sensor
4140 *
4141 * Support for a Průša (or other) IR Sensor to detect filament near the
4142 * extruder
4143 * and make loading more reliable. Suitable for an extruder equipped
4144 * with a filament
4145 * sensor less than 38mm from the gears.
4146 *
4147 * During loading the extruder will stop when the sensor is triggered,
4148 * then do a last
4149 * move up to the gears. If no filament is detected, the MMU2 can make
4150 * some more attempts.
4151 * If all attempts fail, a filament runout will be triggered.
4152 */
4153 // #define MMU_EXTRUDER_SENSOR
4154 #if ENABLED(MMU_EXTRUDER_SENSOR)
4155     #define MMU_LOADING_ATTEMPTS_NR 5 // max. number of attempts to load
4156     filament if first load fail
4157 #endif
4158
4159 #endif
4160
4161 // #define MMU2_DEBUG // Write debug info to serial output
4162
4163 #endif // HAS_PRUSA_MMU
4164
4165 /**
4166 * Advanced Print Counter settings
4167 */
4168 #if ENABLED(PRINTCOUNTER)
4169     #define SERVICE_WARNING_BUZZES 3
4170     // Activate up to 3 service interval watchdogs
4171     // #define SERVICE_NAME_1 "Service S"
4172     // #define SERVICE_INTERVAL_1 100 // print hours
4173     // #define SERVICE_NAME_2 "Service L"
4174     // #define SERVICE_INTERVAL_2 200 // print hours
4175     // #define SERVICE_NAME_3 "Service 3"
4176     // #define SERVICE_INTERVAL_3 1 // print hours
4177 #endif
4178
4179 // @section develop
4180
4181 //
4182 // M100 Free Memory Watcher to debug memory usage
4183 //
4184 // #define M100_FREE_MEMORY_WATCHER
4185
4186 //
4187 // M42 - Set pin states
4188 //
4189 // #define DIRECT_PIN_CONTROL
4190
4191 //
4192 // M43 - display pin status, toggle pins, watch pins, watch endstops &
4193 * toggle LED, test servo probe
4194 //
4195 // #define PINS_DEBUGGING
4196

```

```
4191 // Enable Marlin dev mode which adds some special commands
4192 // #define MARLIN_DEV_MODE
4193
4194 #if ENABLED(MARLIN_DEV_MODE)
4195 /**
4196 * D576 - Buffer Monitoring
4197 * To help diagnose print quality issues stemming from empty command
4198 buffers.
4199 */
4200 // #define BUFFER_MONITORING
4201 #endif
4202 /**
4203 * Postmortem Debugging captures misbehavior and outputs the CPU status and
4204 backtrace to serial.
4205 * When running in the debugger it will break for debugging. This is useful
4206 to help understand
4207 * a crash from a remote location. Requires ~400 bytes of SRAM and 5Kb of
4208 flash.
4209 */
4210 // #define POSTMORTEM_DEBUGGING
4211
4212 /**
4213 * Software Reset options
4214 */
4215 // #define SOFT_RESET_VIA_SERIAL           // 'KILL' and '^X' commands will
4216 soft-reset the controller
4217 // #define SOFT_RESET_ON_KILL             // Use a digital button to soft-
4218 reset the controller after KILL
4219
```